Facultad 4

Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas

Título:

“Sistema Tutorial Inteligente para la Plataforma educativa XAUCE ZERA”

Autor: Carlos Ariel Rojas Lugones

Tutores:

MSc. Yuniesky Coca Bergolla

Ing. Sandy Nuñez Padrón

La Habana, 2018

“Año 60 de la Revolución”

Declaración de autoría

Declaro que soy el único autor del trabajo “Sistema Tutorial Inteligente para la Plataforma educativa XAUCE ZERA” y autorizo a la Facultad 4 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ___ días del mes de _____ del año 2018.

_____

Firma del Autor

Carlos Ariel Rojas Lugones

_____ _____

Firma del Tutor Firma del Tutor

MSc. Yuniesky Coca Bergolla Ing. Sandy Nuñez Padrón

Dedicatoria

A mi abuelo Víctor Cayo Lugones Pérez, donde quiera que esté, por darme fuerza en los momentos difíciles con la imagen de su ejemplo y cariño eternos.

Agradecimientos

A mi madre, Iris Magalys Lugones Bernal, la luz de mi vida, mi ángel de la guarda, que ha soportado valientemente la distancia estos cinco años y siempre ha estado en las buenas y en las malas, que confió siempre en mi esfuerzo, capacidad y compromiso con este momento de la vida. Yo soy quien me siento orgulloso de ella, gracias preciosa.

A Juan José González Salas, quien ha sido más que un amigo, un hermano, de quien puedo decir tantas cosas, pero que se quedarían pequeñas en tan poco espacio, intentaré resumirlo en lo que puedo llamar una amistad inquebrantable, indestructible y duradera.

A mi amigo Sandy Nuñez Padrón, devenido tutor, por soportarme estos cinco años, que no lo hace cualquiera. Por haber sido mi guía fundamental desde que entré en primer año a esta universidad, por esos tiempos de radio, televisión, ACM, estudio y proyectos. Por haberme mantenido tenso desde el primer hasta el último día de quinto año, lo cual me permitió enfocarme en lo importante, y presentar el resultado de este trabajo. Gracias Sandy.

A mi tutor Yuniesky Coca Bergolla, a quien hoy puedo considerar mi amigo, por haberme acompañado en este gran reto que es hoy una realidad, por haber sido un profesor y tutor ejemplar e intachable. Por tener siempre una enseñanza a mano y demostrarme que la vida se toma con calma.

A mi compañero Yasmany Sánchez Aguila, por haber pintado una sonrisa en mi rostro cuando se me había caído el mundo encima, por haber resistido conmigo tantos momentos duros de la vida, por haber entendido que había una sola prioridad: graduarme.

A todos mis profesores, pues todos aportaron a mi formación profesional y personal. De todos aprendí de una forma u otra y dejan una huella imborrable en mi persona.

A mis compañeros de clase, de residencia, de carrera… pasarán los años y los recordaré siempre con una sonrisa.

A los demás profesores de la Facultad 4, que, aunque no recibí clases de la gran mayoría, siempre me trataron con cariño y alegría.

A todos mis amigos, los de aquí y los de allá, que no mencionaré nombres, porque son muchos y para que luego no me digan que me olvidé de alguno… ellos saben bien que ocupan un espacio en mi corazón.

A todos los que siempre estuvieron pendientes de mí. A los que siempre se preocuparon y ocuparon. Al resto de mi familia, a mis abuelas, Nereida y Miriam, a mis padres, Joaquín y Papito. A Esther Cristina y Mario Lázaro, también a Serguei González García, los tres han sido como padres para mí.

A los que lamentablemente ya no están, pero que están más presentes que nunca.

A los que intentaron apagarme, pero que lograron encenderme más. A los ausentes, a los que no pasaron la prueba del tiempo y a los que conocí para no ser como ellos.

A mi país y a mi universidad, por haberme permitido hacer realidad el sueño de estudiar la carrera que siempre deseé.

Muchas gracias.

Carlos Ariel Rojas Lugones.

Resumen

Las Tecnologías de la Información y la Comunicación han revolucionado la forma en que los seres humanos establecen sus relaciones industriales, comerciales, sociales y económicas. Uno de los sectores que ha recibido gran impacto ha sido la educación. En el año 1970 se sientan las bases del primer Sistema Tutorial Inteligente, lo cual desencadenó una creciente producción de los mismos para apoyar el proceso de instrucción. En Cuba se vienen realizando grandes esfuerzos para informatizar el sector educativo. El Centro FORTES de la Universidad de las Ciencias Informáticas cuenta con la Plataforma educativa XAUCE ZERA, la cual apoya el proceso de enseñanza-aprendizaje en el pregrado de dicha institución académica. La plataforma carece de mecanismos que permitan soportar el proceso de tutoría y brindar retroalimentación en tiempo real a las necesidades cognitivas de los estudiantes, utilizando lenguaje natural. El objetivo de la presente investigación es desarrollar un Sistema Tutorial Inteligente para la Plataforma educativa XAUCE ZERA que sea capaz de adaptarse a diferentes dominios de conocimiento y que permita incorporar el proceso de tutoría en la misma.

Palabras clave: Sistema Tutorial Inteligente, Inteligencia Artificial, Procesamiento del Lenguaje Natural, Plataforma educativa XAUCE ZERA.

Índice

Introducción 1

Capítulo 1. Fundamentación teórica 7

1.1. Sistemas Tutoriales Inteligentes 7

1.2. Procesamiento de Lenguaje Natural 10

1.3. Formas de Representación del Conocimiento 11

1.4. Plataforma educativa XAUCE ZERA 13

1.5. Estudio de aplicaciones similares 14

1.6. Técnicas, herramientas y tecnologías a utilizar 18

1.6.1. Lenguajes de desarrollo 18

1.6.2. Sistema Gestor de Bases de Datos 21

1.6.3. Frameworks de desarrollo 21

1.6.4. Entorno de desarrollo integrado 22

1.6.5. Herramienta CASE 22

1.6.6. Sistema de Control de Versiones 22

1.6.7. Metodología de desarrollo de *software* 23

1.7. Conclusiones parciales 24

Capítulo 2. Análisis y diseño de la propuesta de solución 25

2.1. Descripción de la propuesta de solución 25

2.2. Modelo de dominio 27

2.3. Formato de definición de redes semánticas 29

2.4. Generación de respuestas y deducción de perfiles cognitivos 31

2.5. Especificación de requisitos 33

2.5.1. Requisitos funcionales 34

2.5.2. Requisitos no funcionales 37

2.6. Descripción de requisitos mediante Historias de Usuario 38

2.7. Arquitectura 40

2.7.1. Patrón arquitectónico 41

2.8. Modelo de diseño 42
2.8.1. Patrones de diseño 42
2.8.2. Diagrama de clases del diseño 44
2.9. Modelo de despliegue 46
2.10. Diseño de la Base de Datos 46
2.10.1. Diagrama entidad-relación 46
2.10.2. Descripción de las tablas de la base de datos 47
2.11. Conclusiones parciales 49
Capítulo 3. Implementación y prueba de la propuesta de solución 50
3.1. Implementación 50
3.1.1. Generación de respuestas 50
3.1.2. Manejo de perfiles cognitivos 51
3.1.3. Notación intermedia basada en comandos 52
3.1.5. Procesamiento del archivo de definición de red semántica 54
3.1.6. Manejo de intenciones en el agente pedagógico 54
3.1.7. Manejo de preguntas pendientes y respuestas esperadas 55
3.1.8. Librerías utilizadas 55
3.1.9. Diagrama de componentes 56
3.2. Pruebas internas 58
3.2.1. Pruebas unitarias 58
3.2.2. Pruebas de aceptación 60
3.3. Conclusiones parciales 62
Conclusiones 64
Recomendaciones 65
Referencias bibliográficas 66

Índice de figuras

Figura 1.1. Arquitectura de un STI (Alhabbash, Mahdi, & Abu Naser, 2016) 9
Figura 1.2. Plataforma educativa XAUCE ZERA 2.0 14
Figura 2.1. Esquema de comunicación estudiante-STI 26
Figura 2.2. Diagrama de Clases del Modelo de Dominio. 28
Figura 2.3. Arquitectura general de la propuesta de solución 41
Figura 2.4. Diagrama de Clases del Diseño correspondiente al módulo pedagógico 45
Figura 2.5. Diagrama de despliegue. 46
Figura 2.6. Diagrama entidad-relación 47
Figura 3.1. Ejemplo de respuestas estáticas definidas en JSON. 51
Figura 3.2. Ejemplo de patrón sintáctico definido en JSON. 53
Figura 3.3. Un ejemplo de definición de red semántica y su representación gráfica. 54
Figura 3.4. Diagrama de componentes del servidor del STI Iris. 57
Figura 3.5. Diagrama de componentes de STIBundle en XAUCE ZERA 57
Figura 3.6. Capturas de pantalla de los resultados de las pruebas unitarias utilizando *django test suite*. 59
Figura 3.7. Resultados de las pruebas unitarias utilizando *django test suite*. .. 60
Figura 3.8. Resultados de las pruebas usando Postman. 60

Índice de tablas

Tabla 1.1 Comparación de los STI estudiados 17
Tabla 1.2. Fases de la metodología AUP en su variante UCI (Rodríguez Sánchez, 2015). 23
Tabla 2.1. Ejemplo de HU 38
Tabla 3.1. Ejemplo de comandos implementados 52
Tabla 3.2. Diseño de caso de prueba “Incluir red semántica” 61

Introducción

Las Tecnologías de la Información y la Comunicación (TIC) han revolucionado la forma en que los seres humanos establecen sus relaciones industriales, comerciales, sociales y económicas. Uno de los sectores que ha recibido gran impacto ha sido la educación, viéndose beneficiada por estos avances, generando nuevas técnicas para almacenar y transmitir el conocimiento (Duarte Correa & Segovia Vega, 2010).

Actualmente la tecnología ha incursionado fuertemente en la educación, tanto así que se han llevado a cabo múltiples investigaciones sobre cómo poder establecer un ambiente de aprendizaje más interactivo que se acople a las necesidades cognitivas de los estudiantes (Durango Herández & Pascuas Rengifo, 2015).

En el año 1947 se crea la primera computadora electrónica, pero la idea de automatizar el proceso de instrucción surge desde incluso antes. En el año 1912, E. L. Thorndike planteaba la idea de crear un material autoguiado o de una enseñanza programada para funcionar de manera automática. Ya en la década de 1950, en su artículo "*The Science of Learning and Art of Teaching*", B. F. Skinner propone mejoras a las técnicas tradicionales de enseñanza a través del uso de lo que entonces se denominaba "*teaching machines*". Esta idea fue evolucionando y su desarrollo permitió impulsar lo que actualmente se conoce como Instrucción Asistida por Computadoras (IAC). (Duarte Correa & Segovia Vega, 2010)

La IAC se refiere al uso de computadoras en la enseñanza, se basa en aplicaciones informáticas que contienen cursos, exámenes comprobatorios, ejercicios complementarios, simulación de modelos y procesos educativos, así como tutoriales y libros interactivos. El desarrollo de este tipo de tecnología se fue dando a la par del crecimiento de los sistemas computacionales. Poco a poco, las universidades fueron adquiriendo computadoras, primero para la investigación y luego como herramientas auxiliares en la enseñanza, ya que simplificaban el trabajo tanto de estudiantes como profesores.

La IAC ha sido objeto de estudios y de investigaciones, obteniendo buenos resultados, tanto al demostrar su efectividad, como en la implementación de nuevas ideas y mejorar las técnicas de enseñanza que se usan en estos sistemas. Esta forma de instrucción recibió una serie de críticas que fueron establecidas bajo la base de que el estudiante carecía de iniciativa propia o era muy limitada, no era posible utilizar del lenguaje natural en las respuestas, los sistemas eran demasiado rígidos pues su comportamiento estaba preprogramado y no poseían conocimiento real. (Fernández, Server, & Carballo, 2006)

La Inteligencia Artificial (IA), también conocida como inteligencia computacional, es una rama de las ciencias de la computación que surge a finales de la década del 50 y sus técnicas progresaron de forma vertiginosa desarrollando diversos sectores sociales. La IAC recibe un gran impacto de las técnicas de la IA, se facilitan y mejoran los procesos de instrucción y se garantiza mejor eficiencia en los mismos (Duarte Correa & Segovia Vega, 2010).

Ya en el año 1970, Carbonell toma la iniciativa de aplicar las técnicas de la IA a los sistemas de IAC en su artículo "*AI in CAI: An Artificial Intelligence Approach To Computer Assisted Instruction*" (Carbonell, 1970); a él se le atribuye la creación del primer Tutor Inteligente: nombrado SCHOLAR[1]. El término Sistema Tutorial Inteligente (STI) es definido posteriormente por Brown.

Los STI usan la Inteligencia Artificial para incorporar conocimiento y lograr adquirir técnicas educativas para impartir una materia. De esta forma, logran controlar el nivel de aprendizaje de cada uno de los estudiantes y así se garantiza una instrucción personalizada. Dichos sistemas poseen un alto nivel de especialización en el contenido que imparten, convirtiéndose en sistemas expertos. (Duarte Correa & Segovia Vega, 2010) Sus múltiples aplicaciones se enmarcan en un amplio rango de disciplinas como la física, matemática, algoritmia, lingüística, ingeniería, entre otros.

En Cuba se vienen realizando grandes esfuerzos para informatizar la sociedad y de esta forma desarrollar diferentes e importantes sectores sociales, entre ellos

[1] Tutor inteligente utilizado en enseñanza de la geografía de América del Sur

el sector educativo. Como resultado de la batalla de ideas, surge la Universidad de las Ciencias Informáticas (UCI), en el año 2002, institución académico-productiva que muy pronto se vincula a los esfuerzos nacionales por mejorar el sector educativo.

La UCI cuenta con una red de centros de desarrollo, entre los cuales se encuentra el Centro de Tecnologías para la Formación (FORTES), el cual desarrolla diversos productos para la formación y teleformación. Uno de los productos que desarrolla FORTES es la Plataforma educativa XAUCE ZERA, la cual es utilizada para apoyar el proceso de enseñanza-aprendizaje en el pregrado de dicha institución académica. En esta plataforma se encuentran disponibles diversos cursos en línea, correspondientes a las diferentes asignaturas de los planes de estudio de las carreras que se imparten en la universidad.

La Plataforma educativa XAUCE ZERA ofrece interactividad entre los estudiantes y los contenidos educativos mediante foros de discusión, actividades (tareas, ejercicios, cuestionarios, exámenes), etcétera, favoreciendo el aprendizaje en línea (*e-learning*). Estos foros son herramientas que posibilitan, con cierto grado de flexibilidad, crear una comunicación multidireccional (entre todos: docente y estudiantes), de tipo asincrónica con retroalimentación diferida.

En el caso de la Plataforma educativa XAUCE ZERA, los foros son creados por los profesores, asociándolos a determinadas temáticas y sujeto a su voluntad: el profesor crea el foro si lo considera necesario. En estos foros virtuales, al igual que en los debates de las clases presenciales, los estudiantes se comunican, intercambian experiencias e ideas, formulan preguntas, exponen situaciones, responden preguntas, sintetizan pensamientos, reflexionan y cuestionan, todo ello con la intervención del docente que promueve, apoya y retroalimenta los diálogos.

Dado el caso en que un estudiante desee realizar preguntas y plantear dudas en los foros de discusión, debe esperar a que el profesor le dé respuesta y esto no siempre sucede en un lapso de tiempo corto. Esto sucede debido a que el profesor no está conectado a la plataforma las 24 horas del día, sino en horarios

determinados, lo que provoca que no se favorezca la atención diferenciada en tiempo real. Otra de las limitantes que posee la plataforma, es que no es posible utilizar el lenguaje natural al responder ejercicios ni exámenes comprobatorios.

Por lo antes planteado se puede afirmar que la Plataforma educativa XAUCE ZERA carece de mecanismos que permitan soportar el proceso de tutoría y brindar retroalimentación en tiempo real a las necesidades cognitivas de los estudiantes, utilizando lenguaje natural.

Por eso, se plantea como **problema de investigación:** ¿Cómo incorporar el proceso de tutoría en la Plataforma educativa XAUCE ZERA?

Se define como **objeto de estudio:** Sistemas Tutoriales Inteligentes aplicados a plataformas educativas.

Este objeto de estudio se enmarca en el **campo de acción:** Proceso de tutoría en la Plataforma educativa XAUCE ZERA.

Para resolver el problema planteado, se establece como **objetivo general:** Desarrollar un Sistema Tutorial Inteligente para la Plataforma educativa XAUCE ZERA que sea capaz de adaptarse a diferentes dominios de conocimiento.

Para lograr el cumplimiento del objetivo general, se plantean los siguientes **objetivos específicos:**

- Elaborar el marco teórico de la investigación mediante el estudio del estado del arte acerca de las tendencias, las herramientas y las estrategias actuales en el proceso de desarrollo de los Sistemas Tutoriales Inteligentes en plataformas educativas.

- Realizar el análisis, diseño e implementación de un Sistema Tutorial Inteligente para la Plataforma educativa XAUCE ZERA que sea capaz de adaptarse a varios dominios y que contenga características de sistemas similares.

- Definir un formato para representar el conocimiento sobre diversos dominios y que utilizará el sistema diseñado.

- Realizar pruebas de calidad de *software* a la propuesta de solución para verificar el correcto funcionamiento de la misma.

Posibles resultados:

Una vez finalizado el presente trabajo de diploma, se tendrá como resultado un Sistema Tutorial Inteligente para la Plataforma educativa XAUCE ZERA que podrá adaptarse a diferentes dominios de conocimiento, será capaz de recomendar bibliografía a los estudiantes, realizará preguntas de control a los mismos y podrá ofrecer definiciones de conceptos asociados al dominio que cubre. También se tendrá un formato que permitirá representar el conocimiento necesario para el sistema que se propone, mediante el cual los profesores podrán incorporar conocimiento al mismo.

Métodos teóricos:

Histórico-lógico: Se empleó para realizar un estudio de la evolución histórica de los conceptos relacionados con la investigación, entre ellos los conceptos de Inteligencia Artificial y Sistema Tutorial Inteligente.

Analítico-sintético: Se empleó en la realización del estudio del estado del arte de los Sistemas Tutoriales Inteligentes, además para investigar acerca de otras aplicaciones o soluciones similares y de los lenguajes de programación y metodologías de desarrollo de *software* existentes.

Métodos empíricos:

Observación: Permitió extraer información de los objetos observados, específicamente la Plataforma educativa XAUCE ZERA, logrando caracterizarla.

Estructura capitular:

Capítulo 1: Fundamentación teórica. Se realiza un estudio sobre la Inteligencia Artificial, los Sistemas Tutoriales Inteligentes, la Plataforma educativa XAUCE ZERA. Se presenta un estudio del estado del arte. Además, se exponen las técnicas, herramientas y tecnologías a utilizar en el desarrollo de la propuesta.

Capítulo 2: Análisis y diseño de la propuesta de solución. Se realiza el diseño del formato mediante el cual se deberán codificar las redes semánticas aceptadas por el sistema propuesto. Se efectúa el levantamiento de requisitos funcionales y no funcionales de la solución propuesta. Se definen las historias de usuario, así como otros artefactos generados por la metodología de *software* seleccionada.

Capítulo 3: Implementación y prueba de la propuesta de solución. Se realiza una evaluación de los parámetros expuestos en el diseño para una correcta implementación, atendiendo a la descripción de las principales funcionalidades. Además, se le realizan pruebas a la solución propuesta, comprobando la correcta ejecución de las funcionalidades.

Capítulo 1. Fundamentación teórica

El presente capítulo aborda el conjunto de conceptos y fundamentos que constituyen el marco teórico relacionado con el objeto de estudio definido en la investigación. Se destacan las principales características de los elementos asociados a la temática a investigar, así como el entorno nacional e internacional donde se desarrollan. Se exponen la metodología y las principales herramientas que sirven de apoyo para la búsqueda de una solución a la problemática planteada.

1.1. Sistemas Tutoriales Inteligentes

Las tecnologías asociadas a la IA fueron incorporadas en la educación y la enseñanza en 1970 en diferentes marcos de trabajo como es el caso de los STI. (Alhabbash, Mahdi, & Abu Naser, 2016)

En años recientes, ha habido un auge en el campo de los STI, estos han llamado mucho la atención de los investigadores, lo cual ha promovido el surgimiento de múltiples investigaciones y propuestas en diferentes dominios de conocimiento y renglones de la sociedad.

La necesidad real de los STI en el campo de la educación, queda plasmada en la siguiente afirmación de Barbhuiya y otros autores: “En un aula real, un profesor experto humano, que es rico en conocimiento del dominio que imparte, explica los conceptos a los estudiantes usando varias herramientas, luego, las evaluaciones conducen a probar el grado de maestría del estudiante sobre un tema dado. Durante estas interacciones, el profesor intenta modelar y evaluar el comportamiento individual de los estudiantes y de acuerdo a ello decide la estrategia pedagógica. Un STI debiera intentar imitar al mejor de los maestros mientras pone mayor esfuerzo en la atención individualizada, porque es ahí donde los profesores humanos fallan o se quedan cortos debido al tiempo y otras limitaciones”. (Barbhuiya, Mustafa, & Jabin, 2011)

Los STI ya no son un área de investigación joven, de hecho, han revolucionado el *e-learning*, la instrucción curricular y el entrenamiento en lugares de trabajo. En las últimas décadas, se han visto diferentes enfoques en el campo de los STI, que llevan a diferentes direcciones y, como resultado de esto, numerosos STI se

han desarrollado hasta la fecha. Sin embargo, la mayoría han sido estudiados en entornos de investigación y solo unos pocos han demostrado ser exitosos en entornos académicos reales para un gran número de estudiantes. Esto se debe fundamentalmente a que el fenómeno del aprendizaje humano es muy complejo y en sí mismo constituye un campo de actividad investigativa que constantemente se ha mantenido activo a lo largo de la historia de la humanidad. (Barbhuiya, Mustafa, & Jabin, 2011)

El análisis realizado del concepto de STI visto desde el enfoque de diferentes autores (Wolf, 1984), (VanLehn K. , 1988), (Burns & Capps, 1988), (Guardia Robles, 1993), (Giraffa, Nunes, & Viccari, 1997), (Crawford, 2013), (Alhabbash, Mahdi, & Abu Naser, 2016), permitió extraer similitudes entre los mismos, lo cual ofrece una visión más amplia sobre el concepto. En la presente investigación, se asume que un STI es un sistema de *software* que utiliza técnicas de IA para representar el conocimiento e interactúa con los estudiantes para enseñárselo, simulando a un profesor humano, que debe ser capaz de deducir la cercanía del conocimiento del estudiante con el que está representado en el sistema y, además, debe poder reducir dicha distancia; debe tener la capacidad tanto para resolver el problema que se le pueda presentar a un estudiante como de explicar cómo lo resolvió; el sistema realiza preguntas al estudiante, analiza su respuesta y ofrece una instrucción y retroalimentación personalizada; construye un perfil para cada estudiante y estima el grado de dominio del mismo sobre una temática dada.

Arquitectura

Una arquitectura típica de un STI consiste en una base de conocimientos (BC), el modelo del estudiante, el módulo pedagógico y la Interfaz Gráfica de Usuario (GUI) que permita comunicación con el usuario. (Dede, 1986)

Alhabbash y colectivo de autores muestran en la Figura 1.1 una vista gráfica de esta arquitectura y cómo se relacionan sus componentes. (Alhabbash, Mahdi, & Abu Naser, 2016)

La BC, también llamada “modelo de dominio”, es el conocimiento relacionado con el tema que se le enseñará al estudiante. Es la fuente de conocimiento para los estudiantes (Alhabbash, Mahdi, & Abu Naser, 2016)

El modelo del estudiante es un modelo que representa los procesos que intervienen en el conocimiento como la resolución de problemas, la extracción de información, el aprendizaje a partir de errores, es decir, el nivel de aprendizaje del estudiante y su patrón de aprendizaje. (Alhabbash, Mahdi, & Abu Naser, 2016)

El módulo pedagógico, también llamado modelo de tutor es el conocimiento acerca de cómo enseñar a los estudiantes, así como el conocimiento relacionado con las reglas usadas para enseñar un dominio en particular. (Alhabbash, Mahdi, & Abu Naser, 2016)

El modelo de GUI es la forma de interacción del estudiante con el STI, que permite la comunicación y los diálogos con el mismo. (Alhabbash, Mahdi, & Abu Naser, 2016)

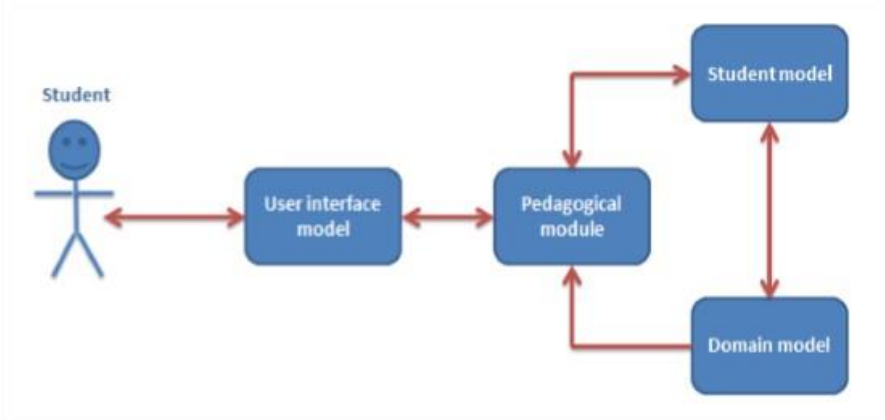

Figura 1.1. Arquitectura de un STI (Alhabbash, Mahdi, & Abu Naser, 2016)

En su conferencia del año 2011, Barbhuiya y otros autores plantean que el éxito de un STI depende en gran medida de la habilidad de entender los factores que inciden en el aprendizaje humano y modelarlos mediante los cuatro módulos básicos de este tipo de sistemas: el módulo de dominio, el módulo pedagógico, el módulo estudiante y la GUI. Un STI puede ser muy rico en su dominio, pero adoptar una política pobre al proveer retroalimentación; mientras que otro puede ser bueno al proveer retroalimentación, pero pone muy poca atención en los estados afectivos del estudiante, así como en la instrucción individualizada. Un STI ideal debiera desarrollarse teniendo en mente todos esos aspectos y, por lo tanto, la investigación en este campo debe ser multidisciplinaria por naturaleza. Sin embargo, la mayoría de los STI existentes carecen de uno o varios de estos aspectos. (Barbhuiya, Mustafa, & Jabin, 2011)

Evolución

Los STI han ido evolucionando a medida que transcurre el tiempo y cambia la tecnología. En 2004, González muestra la evolución de los STI de acuerdo a sus generaciones (González, 2004):

- **STI de Primera generación:** Tutores Basados en Restricciones (*Constraint Based Tutors*). Se centran en el estado de la interface, en relación a la información que se muestra u oculta, gráficos, ayudas, etcétera.
- **STI de Segunda generación:** Tutores basados en modelos cognitivos (*Model-Tracing Tutors* / *Cognitive Tutors*). Se centran en las acciones del estudiante y las reglas que generan la solución correcta a una tarea propuesta basada en un modelo mental del estado cognitivo (y de la memoria de trabajo) del alumno.
- **STI de Tercera generación:** Tutores basados en diálogo-lenguaje natural (*Dialogue Based Tutors*). Se centran en la comunicación con el alumno a través del dialogo. Las ayudas, las explicaciones a los ejercicios, críticas y discusiones sobre un tema o problema se realizan por medio del dialogo entre el sistema y el alumno.

Como se ha afirmado, la Plataforma educativa XAUCE ZERA precisa de un STI que permita realizar diálogos en lenguaje natural, es decir, de tercera generación.

1.2. Procesamiento de Lenguaje Natural

Según Liddy, el Procesamiento de Lenguaje Natural (PLN: *Natural Language Processing*) es un enfoque computarizado para analizar texto que está basado en un conjunto de teorías y tecnologías. Al ser una rama muy activa de investigación y desarrollo, no existe una definición única ampliamente aceptada que pueda satisfacer a todos. Sin embargo, Liddy ofrece su propia definición: “El PLN es un rango (motivado teóricamente) de las técnicas computacionales para analizar y representar textos que ocurren de forma natural en uno o más niveles de análisis lingüístico con el propósito de lograr el procesamiento del lenguaje humano para un rango de tareas o aplicaciones”. (Liddy, 2001)

El PLN tiene dos divisiones diferentes: procesamiento del lenguaje y generación del lenguaje. La primera división se refiere al análisis del lenguaje con el propósito de producir una representación significativa del mismo, mientras que la segunda se refiere a la producción del lenguaje a partir de una representación. La tarea del procesamiento del lenguaje es el equivalente al rol de rector/receptor (del mensaje) mientras que la generación del lenguaje es el equivalente al rol de escritor/emisor. (Liddy, 2001)

Mientras continúan progresando las investigaciones en esta área, algunos fenómenos recientes como el *big-data*, las tecnologías móviles, las redes sociales y los Cursos Abiertos Masivos en Línea (MOOCs: *Massive Open Online Courses*) han resultado en la creación de nuevas oportunidades y desafíos de investigación. Algunas aplicaciones comerciales incluyen asistentes de escritura y ambientes de instrucción en línea. (Litman, 2016)

Los avances en el PLN y las tecnologías educacionales, así como la disponibilidad de grandes cantidades de texto relevante en el campo educativo y también de datos del habla, han llevado a un interés creciente en usar el PLN para dirigir las necesidades de profesores y estudiantes. (Litman, 2016)

1.3. Formas de Representación del Conocimiento

En la revisión bibliográfica realizada, no se encontró una definición formal ampliamente aceptada del concepto de Forma de Representación del Conocimiento (FRC).

Bello Pérez y colectivo de autores plantean que la cuestión básica de la representación del conocimiento es el desarrollo de una notación suficientemente precisa mediante la cual se pueda representar el mismo, llaman FRC a esa notación. Como no existe una FRC general capaz de ser usada con éxito en todo tipo de aplicaciones, las formas disponibles están limitadas. Para un problema en particular, estos autores señalan diversos criterios que deben ser tomados en cuenta (Bello Pérez, García Valdivia, García Lorenzo, & Reynoso Lobato, 2002):

- Debe permitir describir el dominio de una manera natural, reflejando tanto como sea posible la estructura de los objetos, los hechos y las relaciones entre ellos.
- Debe aceptar conocimiento empírico, teórico o heurístico, y combinar el conocimiento declarativo con el procedimental, de acuerdo con los requerimientos de la aplicación.
- Debe permitir estratificar el conocimiento de acuerdo con su significado y funciones.
- Debe permitir que el conocimiento de los expertos se represente fácilmente.
- Los expertos en la aplicación deben poder comprender el conocimiento que está almacenado.
- El conocimiento almacenado se debe poder usar con efectividad.

Una **red semántica** consta de puntos llamados nodos, conectados por enlaces llamados arcos o aristas que describen las relaciones entre los nodos. Los nodos representan objetos, conceptos o eventos. Los arcos pueden definirse de varias formas, dependiendo de la clase de conocimiento representado. Frecuentemente se ilustra una red semántica como un grafo etiquetado dirigido. Esta FRC enfatiza en la conectividad del conocimiento análogamente a la construcción de un diccionario. Cada palabra (concepto) es definida en términos de otras, creando una estructura enlazada que interrelaciona virtualmente cualquier objeto. (Bello Pérez, García Valdivia, García Lorenzo, & Reynoso Lobato, 2002)

Una **ontología** (como concepto de las Ciencias de la Computación) es un tipo especial de objeto de información o artefacto computacional que se basa en el precepto de la IA conocido como “lo que ‘existe’ puede ser representado”. Son un modelo formal de la estructura de un sistema, por ejemplo, las entidades y relaciones relevantes que emergen de su observación y que son útiles para los propósitos de quienes la modelan. La columna vertebral de una ontología consiste en una jerarquía de generalización/especialización de conceptos. Otros autores la definen como una “especificación formal de una conceptualización compartida”, lo cual sugiere que dicha conceptualización debe expresar un punto de vista compartido entre diferentes partes, es decir, un consenso más que un

punto de vista individual, por otro lado, dicha especificación debiera estar expresada en un formato leíble por computadoras. (Staab & Studer, 2009)

Las ontologías son hoy día la FRC más completa existente, pero debido a su gran complejidad de elaboración, no están ampliamente extendidas, su uso implica la introducción de un problema práctico en el desarrollo de los sistemas. La creación de uno de estos artefactos requiere de equipos interdisciplinarios de expertos de diferentes ramas del conocimiento. Ambas FRC (redes semánticas y ontologías) se acoplan naturalmente con el PLN, lo cual soporta la valoración realizada sobre las mismas.

1.4. Plataforma educativa XAUCE ZERA

XAUCE ZERA (Figura 1.2) se caracteriza por innovadora, flexible, fácil de usar, interactiva y adaptable, siendo capaz de guiar el proceso de enseñanza-aprendizaje en empresas, organizaciones, comunidades e instituciones de cualquier nivel de enseñanza a partir de un conjunto poderoso de herramientas centradas en los aprendices y ambientes de aprendizaje colaborativo, que le dan poder, tanto a la enseñanza como al aprendizaje.

Otras de sus características son su accesibilidad desde cualquier dispositivo móvil y su soporte de contenidos interactivos enriquecidos con elementos psicológicos y didácticos, así como recursos multimedia a partir de plantillas previamente definidas que se entremezclan con documentos, archivos, presentaciones y otros materiales junto a foros de discusión, cuestionarios y disímiles tipologías de ejercicios que pueden ser evaluados de forma automática, por coevaluación o directamente por el profesor.

La Plataforma educativa XAUCE ZERA brinda soporte a los MOOCs y asegura que no haya limitaciones lingüísticas para aprender en línea dando soporte a varios idiomas. Así mismo, presenta una estructura multiorganizativa que permite ser utilizada por diferentes instituciones. Por sus funcionalidades, brinda soporte a las necesidades tanto de clases pequeñas, como de grandes organizaciones debido a su flexibilidad y escalabilidad.

También aporta a la enseñanza y el aprendizaje con tecnología educativa innovadora que ayuda a los profesores a adaptarse a las nuevas normas y

personalizar el aprendizaje ofreciendo experiencias nuevas y emocionantes de aprendizaje digital y garantizando que todos los estudiantes tengan la oportunidad de desarrollar todo su potencial.

Cuenta con un editor de texto enriquecido que permite darle el formato al texto según el gusto y la intención del usuario, así como añadir recursos multimedia haciendo que la creación de materiales de cursos de alta calidad para los estudiantes sea más fácil.

Los recursos en XAUCE ZERA están constituidos por páginas, cuestionarios, actividades y recursos educativos. Permite además definir claves de evaluación (también conocidas como rúbricas) para las actividades evaluativas que pueden ser evaluadas de forma automática, por el profesor o por coevaluación.

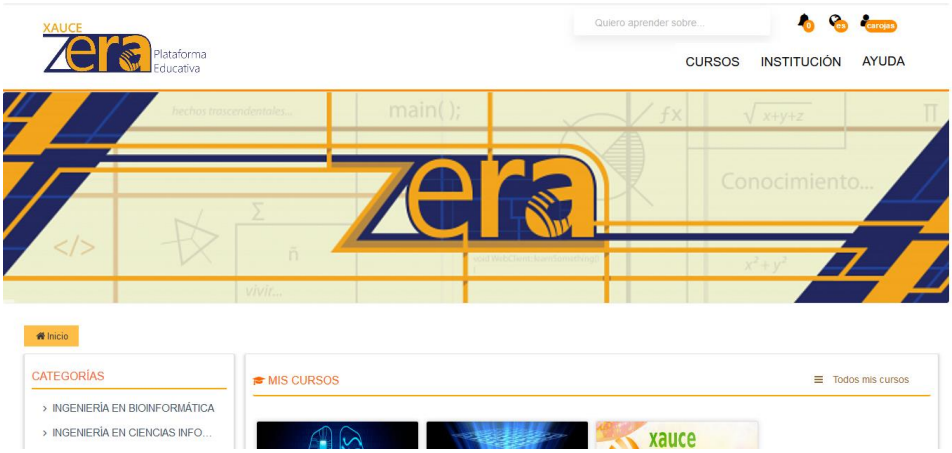

Figura 1.2. Plataforma educativa XAUCE ZERA 2.0

1.5. Estudio de aplicaciones similares

En el campo de los STI existen diversas aplicaciones similares, estudiarlas permite extraer características comunes, así como ventajas y desventajas. A continuación, se expone el estudio realizado.

Scholar

Este sistema estaba diseñado para enseñar la Geografía de América del Sur, abarcando conceptos como país, estado y población; se basó en una aplicación conocida como ELIZA (un *bot* conversacional) (Wenger, 1987). Es un sistema de instrucción que usa iniciativa mixta, utiliza bases de datos modeladas por expertos humanos que le brindan conocimiento. Dicho conocimiento acerca de

cualquier tema está dispuesto en forma de una red semántica estática de hechos, conceptos y procedimientos, lo cual tiene la desventaja de que el sistema posee conocimiento estático y por tanto queda obsoleto rápidamente. Otra de las desventajas de *Scholar* es que el diálogo no es anticipado y depende de las respuestas, preguntas y peticiones del estudiante; además, resulta incómoda la comunicación ya que el diálogo escrito debe comenzar y terminar con un asterisco (*) en el caso del estudiante. Su interacción con el usuario se basa en un diálogo escrito, en forma de chat. Utiliza como lenguaje un subconjunto del inglés, viéndose limitado por las primitivas capacidades sintácticas del sistema. (Wenger, 1987)

SQL-Tutor

SQL-Tutor, desarrollado por Mitrovic, enseña y explica a los estudiantes la forma de escribir consultas en una base de datos (BD) relacional a través de diferentes lecciones básicas y, además, el estudiante introduce la consulta al sistema y este la analiza para hallar errores y defectos. Dependiendo de los errores, el sistema le ofrece un conjunto de notas y consejos mostrándole un texto corto que describe el error y cómo solucionarlo. (Mitrovic, 1998)

Sistema Tutorial de física Andes

El Sistema tutorial de física Andes (*Andes physics tutoring system*), es un sistema que enseña a los estudiantes diferentes formas de resolución de problemas y ejercicios de física, guiándolos mientras estos resuelven los mismos. (VanLehn, y otros, 2005).

Andes permite a los estudiantes dibujar diagramas, entrar ecuaciones y definir variables con la misma libertad que cuando usan el papel. Sobre la marcha, el sistema le comunica al estudiante si su entrada es correcta o incorrecta, marcándola en verde o rojo. (The Andes Physics Tutor, 2017)

MetaTutor

MetaTutor es un STI que enseña a los estudiantes cómo generar pequeños sub-objetivos de un problema mientras aprenden temas relacionados con la ciencia y a generar la retroalimentación necesaria. (Barbhuiya, Mustafa, & Jabin, 2011)

El éxito de MetaTutor levanta esperanzas de lograr independencia del dominio porque su arquitectura separa el módulo de dominio del resto de las partes. (Barbhuiya, Mustafa, & Jabin, 2011)

AnimalWatch

AnimalWatch, un STI diseñado con el objetivo de ayudar a los estudiantes a dominar habilidades básicas de cálculo y fracciones. Usa los errores cometidos durante la resolución de problemas para estimar las habilidades del estudiante en cada tema de las matemáticas, y selecciona problemas que el estudiante debería ser capaz de resolver usando los recursos de ayuda integrados. Un enfoque alentador en este estudio es la metodología de evaluación basada en pruebas realizadas al principio y al final. (Barbhuiya, Mustafa, & Jabin, 2011)

ELM-ART tutor

ELM-ART tutor, es un sistema educacional inteligente e interactivo que soporta el aprendizaje de la programación utilizando el lenguaje LISP. Provee todos los materiales de aprendizaje *online* en forma de un libro de texto adaptativo e interactivo. (Barbhuiya, Mustafa, & Jabin, 2011)

PUMP Algebra Tutor

PUMP Algebra Tutor (PAT) fue desarrollado por el *Pittsburg Urban Mathematics Project* (PUMP). Está basado en la *ACT Theory* y en la tecnología de tutores cognitivos. El modelo cognitivo está escrito como un sistema de reglas de producción “si-entonces” que son capaces de generar las múltiples soluciones, así como los pasos típicos dados o perdidos por los estudiantes. (Barbhuiya, Mustafa, & Jabin, 2011)

Comparación de los sistemas estudiados

El análisis realizado (Tabla 1.1) permitió identificar las características fundamentales de los STI más representativos, según la bibliografía consultada. Se evidencia que todos se limitan a un dominio específico, en ningún caso se especifica si pueden o no ser extensibles para otros dominios. En cuanto al modelo del estudiante todos lo realizan de forma diferente, lo cual evidencia que no existe consenso en la mejor forma de lograrlo, pero sí en la necesidad de tenerlo.

Tabla 1.1 Comparación de los STI estudiados

		PROPIEDADES				
		Dominio cubierto	Retroalimentación	Uso de agentes animados	Modelo de estudiante	Otras características
TUTORES	Scholar	Geografía de América del Sur	Inmediata	No		Maneja valores precisos y valores difusos
	SQL-Tutor	Consultas SQL	5 niveles de retroalimentación	No	*CBM for short-term and Overlay for long-term modelling*	
	ANDES	Física	Inmediato	No	Red bayesiana	
	MetaTutor	Cuerpo humano	Verbal	Sí	Hipermedia adaptativa	Aprendizaje autoregulado
	AnimalWatch	Matemáticas	Inmediato	Sí	Modelos ocultos de Markov	
	ELM-ART tutor	Programación en LISP	Inmediato	No	Episódico / Colaborativo (abierto, editable)	Muestra materiales como un libro interactivo y adaptable
	PAT	Álgebra	Inmediato (oportuno)	No	Model tracing and knowledge tracing (*ACT Theory*)	

1.6. Técnicas, herramientas y tecnologías a utilizar

En esta sección se exponen las técnicas, herramientas y tecnologías a utilizar en el desarrollo de la propuesta de solución. Se fundamenta la elección de los lenguajes de programación y otras herramientas informáticas, así como la metodología de desarrollo de *software*.

1.6.1. Lenguajes de desarrollo

En la actualidad existe una gran diversidad de lenguajes de programación, cada uno de los mismos posee una utilidad propia que le permite ser aplicado a la resolución de ciertos tipos de problemas, aunque muchos de ellos se consideran de propósito general, es decir, que pueden ser aplicados de forma genérica en cualquier situación que se desee.

Lenguajes de desarrollo del lado del servidor

Cada aplicación web tiene asociado un servidor en el cual se procesan las peticiones y se generan las respuestas que posteriormente son enviadas a la estación cliente.

Python es un lenguaje de programación orientado a objetos, que utiliza una sintaxis elegante y legible, disponible bajo licencia de *software* libre, fácil de usar, fácil de extender y multiplataforma. Provee una amplia librería estándar que soporta diversas tareas comunes de programación. (Sitio oficial de Python, 2018)

En su publicación de carácter anual conocida como Radar Tecnológico, la consultora tecnológica *ThoughtWorks*, ofrece su “visión sobre la tecnología y tendencias que dan forma al futuro”. En su volumen 16 (año 2016) plantean que “su facilidad de uso como un lenguaje de programación de propósito general en conjunto con sus fuertes fundamentos matemáticos y de computación científica ha permitido su adopción mayoritaria por las comunidades académicas y de investigación”, y añaden que recientemente, “las tendencias de la industria sobre la popularización de IA y sus aplicaciones, combinada con la madurez de Python 3, han ayudado a traer nuevas comunidades al mundo de Python”. De acuerdo a estos y otros criterios, recomiendan fuertemente la adopción de Python 3. (ThoughtWorks,

Technology Radar, 2016) Adicionalmente, en el volumen 17 (año 2017) ratifica, en su sección de lenguajes y *frameworks,* Python en su versión 3 o posterior como el lenguaje que recomiendan fuertemente adoptar (ThoughtWorks, Technology Radar, 2017).

PHP, acrónimo de "Hypertext Preprocessor", es un lenguaje ampliamente extendido en la web actual. Su código fuente es interpretado por el servidor generando una respuesta HTTP que generalmente contiene un cuerpo definido en HTML. Ampliamente utilizado, de código abierto y de propósito general bajo licencia GPL. Es un lenguaje de scripting adecuado para el desarrollo web y embebido en páginas HTML. El principal objetivo del lenguaje es permitir que los desarrolladores web escriban páginas generadas dinámicamente de forma rápida. (Manual de PHP, 2018)

Entre sus principales características destacan su potencia, alto rendimiento, facilidad de aprendizaje y escasez de consumo de recursos. Es multiplataforma y con capacidad de conexión con la mayoría de los sistemas de gestión de bases de datos actuales. (Manual de PHP, 2018)

Se selecciona este lenguaje para la implementación de las funcionalidades del lado del servidor en la Plataforma educativa XAUCE ZERA debido a que la misma está escrita en dicho lenguaje.

Lenguajes de desarrollo del lado del cliente

HTML (*Hypertext Markup Language*) es el lenguaje de marcado que constituye el núcleo de la World Wide Web (WWW). Originalmente, HTML fue primeramente diseñado para describir documentos científicos. Su diseño general, sin embargo, le ha permitido adaptarse a lo largo de la historia para describir otros tipos de documentos e inclusive aplicaciones. (W3C, 2018)

CSS (*Cascading Style Sheets*) es el lenguaje usado para describir la presentación de las páginas web, lo cual incluye colores, diseño y fuentes. Permite adaptar la presentación a diferentes tipos de dispositivos, como los de pantallas largas, pequeñas e inclusive impresoras. CSS es independiente de HTML y puede ser

usado con cualquier lenguaje basado en XML. La separación de CSS y HTML hace más fácil mantener los sitios, compartiendo estilos a través de diferentes páginas y adaptar páginas a diferentes entornos. (W3C, 2018)

El lenguaje de *scripting*[2] más común en la web es ECMAScript (más conocido como **JavaScript**), que ha sido embebido en entornos de ejecución como los navegadores. Es un lenguaje de programación interpretado que hace más dinámicas las páginas web, permitiendo por ejemplo recargar porciones de las mismas sin tener que refrescarlas completamente, o permitiendo enviar y recibir contenido de dicha página, usando AJAX (*Asynchronous JavaScript And XML*). Detrás de esto, JavaScript permite establecer comunicación entre el navegador y el servidor, haciendo posible, por ejemplo, crear páginas web que incorporen información del entorno del usuario, como la locación, detalles de la libreta de direcciones, entre otros. Esta interactividad adicional permite que las páginas web se comporten como una aplicación tradicional de *software*, lo cual ha permitido a estas páginas ser llamadas como "aplicaciones web". (W3C, 2018)

Estos lenguajes se entremezclan de forma que HTML provee la estructura de la página, CSS define el diseño y la forma en que se presentan los elementos y JavaScript define el comportamiento de los componentes del *Document Object Model* (DOM) de la página web. (W3C, 2018)

Lenguaje de intercambio de datos

El intercambio de datos entre aplicaciones y cómo representarlos en un formato agradable y universal resulta una de las cuestiones de gran interés en la actualidad por parte de la industria de *software*. Un lenguaje para lograr esto es JSON.

JSON (*JavaScript Object Notation*) es un formato ligero de intercambio de datos. Es sencillo de escribir y leer por humanos, como también es fácil de generar e interpretar por las máquinas. Está basado en un subconjunto del lenguaje de programación JavaScript, pero que es completamente independiente de dicho

[2] En el contexto de un navegador, el término *scripting* se refiere al código fuente de un programa escrito en JavaScript que es ejecutado por el navegador cuando una página es descargada, o en respuesta a un evento desencadenado por el usuario. (W3C, 2018)

lenguaje. Además, usa convenciones que le son familiares a los desarrolladores de la familia de C, C++, C#, Java, JavaScript, PERL, Python, entre otros. Estas propiedades hacen de JSON un lenguaje ideal para el intercambio de datos. (Introducing JSON, 2018)

Lenguaje de modelado

El **Lenguaje de Modelado Unificado** (UML: *Unified Modeling Language*) se ha convertido en el lenguaje aceptado universalmente para los planos del diseño de *software*, es una de las mejores herramientas para analizar y diseñar sistemas de *software* orientados a objetos que ofrece un lenguaje común que todo desarrollador debiera conocer. (Larman, 2003)

1.6.2. Sistema Gestor de Bases de Datos

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia *Berkeley Software Distribution* (BSD) y con código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado. PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. (PostgreSQL, Portal en español sobre PostgreSQL, 2018)

1.6.3. Frameworks de desarrollo

Usualmente los lenguajes más usados y potentes están acompañados de *frameworks* de desarrollo también potentes. **Django** es “un *framework* web de alto nivel destinado a Python que alienta el desarrollo rápido y limpio, con un diseño pragmático”. Este *framework* es desarrollado por “desarrolladores experimentados, lo cual se encarga de gran parte de la molestia asociada al desarrollo web”. (Django Documentation, 2018)

Otro *framework*, llamado **Django Rest Framework** y basado en Django, es descrito como “un poderoso y flexible kit de herramientas para construir REST APIs” (Django Rest Framework, 2018).

Se utilizará además el *framework* **Symfony** para la integración de la propuesta de solución con la Plataforma educativa XAUCE ZERA. Este *framework* ha sido ideado para aprovechar las características de PHP, es rápido, flexible y fácil de usar. Es un marco de trabajo de código abierto que está construido utilizando un contenedor de inyección de dependencias. (Eguiluz, 2011)

1.6.4. Entorno de desarrollo integrado

Los IDE (*Integrated Development Environment*) son programas informáticos compuestos por un conjunto de herramientas de programación. Pueden dedicarse a un lenguaje de programación específico o a varios. En cualquiera de los dos casos proveen un marco de trabajo amigable. Están constituidos por un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. (Pérez González & Nuñez Padrón, 2015)

Como IDE se seleccionó **NetBeans** 8.0.2 por ser un entorno de desarrollo abierto, modular y de base estándar (normalizado). En este IDE se pueden desarrollar aplicaciones usando los lenguajes anteriormente planteados (PHP, Python, CSS, HTML, JavaScript). (Sitio oficial de Net Beans, 2018)

1.6.5. Herramienta CASE

Las herramientas CASE (Ingeniería de *Software* Asistida por Computadora), son diversas aplicaciones informáticas destinadas a aumentar la productividad en el ciclo de vida del desarrollo de *software*.

Visual Paradigm es una herramienta CASE que soporta desde la planificación, el análisis y el diseño, hasta la generación de código fuente y la documentación de un proyecto de desarrollo de *software*. Concebida para soportar el ciclo de vida completo del proceso de desarrollo de *software* a través de la representación de todo tipo de diagramas. Constituye una herramienta de *software* libre de probada utilidad para los analistas. (Sitio oficial de Visual Paradigm, 2018)

1.6.6. Sistema de Control de Versiones

Git es un sistema de control de versiones libre, de código abierto y distribuido, diseñado para manejar cualquier tipo de proyecto, desde los pequeños a los grandes con rapidez y eficiencia. Es fácil de aprender y tiene un rendimiento alto. (Sitio oficial de Git, 2018)

1.6.7. Metodología de desarrollo de *software*

AUP (*Agile Unified Process*) es una versión simplificada de la metodología RUP (*Rational Unified Process*) que aplica técnicas ágiles incluyendo el desarrollo dirigido por pruebas, el modelado ágil, la gestión de cambios ágil y la refactorización de la BD para mejorar la productividad. (Rodríguez Sánchez, 2015)

AUP establece, al igual que RUP, cuatro fases que se transcurren de forma consecutiva, aunque agrupadas: Inicio, Elaboración, Construcción y Transición. (Rodríguez Sánchez, 2015)**.**

AUP en su variante UCI (Tabla 1.2) mantiene la fase de Inicio, y se unifican las restantes fases de AUP en una única fase, llamada ejecución. Adicionalmente se añade la fase Cierre. (Rodríguez Sánchez, 2015)

Tabla 1.2. Fases de la metodología AUP en su variante UCI (Rodríguez Sánchez, 2015).

Fase	Objetivo
1. Inicio	Se planifica el proyecto, se realiza un estudio del negocio y del alcance del proyecto, se realizan estimaciones del tiempo, esfuerzo y costo y se decide si se ejecuta o no el proyecto.
2. Ejecución	Se desarrolla el *software*, incluyendo el ajuste de los planes del proyecto, considerando los requisitos y la arquitectura. Se modela el negocio, se obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y libera el producto.
3. Cierre	Se analizan los resultados del proyecto y se realizan las actividades formales de cierre del proyecto.

1.7. Conclusiones parciales

El análisis de estado del arte permitió identificar los módulos tradicionales de un STI, la necesidad de representar el conocimiento usando la notación adecuada y de efectuar PLN.

A partir del estudio de las aplicaciones similares y del análisis de sus principales características, se determinó que ninguno de los STI estudiados constituye una solución viable para resolver el problema planteado. Sin embargo, se identificó un conjunto de funcionalidades a tomar en cuenta para el desarrollo de la propuesta de solución.

El conjunto de herramientas y tecnologías presentadas permitirá llevar a cabo el desarrollo del sistema. Se seleccionó AUP en su variante UCI como metodología de desarrollo y Visual Paradigm 8.0 como herramienta CASE. Los lenguajes de desarrollo a utilizar son UML como lenguaje de modelado, PHP 7.0.8 y Python 3.5.0 como lenguajes de programación del lado del servidor y JSON como lenguaje de intercambio de datos, así como HTML 5, CSS 3 y JavaScript como lenguajes de desarrollo del lado del cliente. Los frameworks a utilizar serán Symfony 2.7.16 y Django 1.11.6, destinados a PHP y a Python respectivamente, así como el framework Django Rest Framework 3.7.7 para el desarrollo de la REST API. El IDE seleccionado es NetBeans 8.0.2 y se empleará PostgreSQL 9.4 como sistema gestor de bases de datos.

Capítulo 2. Análisis y diseño de la propuesta de solución

El progreso de la presente investigación precisa definir las características y funcionalidades que debe poseer el sistema a desarrollar. Con ese objetivo, se describe en este capítulo la propuesta de solución. Se expone el modelo de dominio, la especificación y descripción de requisitos, el patrón arquitectónico, así como otras herramientas y artefactos de la metodología seleccionada.

2.1. Descripción de la propuesta de solución

Para darle solución al problema planteado, el STI que se propone, nombrado “Iris”, permitirá que el profesor a cargo de un curso dentro de la Plataforma educativa XAUCE ZERA pueda incluir una red semántica y asociarla a dicho curso. La red semántica importada representará el conocimiento que dominará el STI Iris sobre el contenido del curso, estará definida en un formato específico (véase 2.3). En la propia definición de la red semántica estarán especificados los nodos más importantes que el tutor inteligente deberá chequear para establecer el perfil cognitivo de cada estudiante.

El perfil cognitivo será la forma en que el STI Iris establecerá el modelo del estudiante, permitiéndole tener creencias en cuanto a cuáles son los contenidos que domina el estudiante y en qué medida los domina. Esta medida estará dada por una distancia para cada nodo importante de la red semántica definida.

El estudiante, por su parte, podrá acceder a la interfaz de comunicación con el STI Iris, que tendrá el funcionamiento y la estética de un *chat*, mediante el cual podrán intercambiar información y conocimiento a través de diálogos en lenguaje natural (Figura 2.1).

Figura 2.1. Esquema de comunicación estudiante-STI

Algunas de las funcionalidades principales que deberá poseer el STI que se describe en la interacción con el estudiante, son las siguientes:

- Al iniciar la primera conversación con un estudiante, el STI Iris saluda y da la bienvenida, además, propone un tema de conversación.
- Al despedirse el estudiante, el sistema se despide y cierra la conversación.
- El sistema podrá responder preguntas relacionadas con su identidad, permitiéndole identificarse ante el estudiante.
- El estudiante podrá en cualquier momento proponer y establecer un tema de conversación, cambiarlo y preguntar por el tema actual, en caso de existir. Podrá además preguntar por los posibles temas de conversación disponibles y proponer que se cambie el tema de conversación.
- El estudiante podrá pedir al sistema que este le ofrezca la definición de un concepto asociado al tema de conversación actual, dado ese caso, el sistema responderá con la definición del concepto si lo posee en su BC.
- El estudiante podrá pedir al sistema que este le recomiende fuentes bibliográficas asociadas a un concepto o contenido, de forma que pueda profundizar en el mismo. El sistema responderá en lenguaje natural devolviendo el listado de las fuentes bibliográficas asociadas al concepto, en caso de existir.
- El sistema podrá en cualquier momento realizar preguntas de control al estudiante con el objetivo de establecer el perfil cognitivo del mismo, pidiéndole que complete definiciones de conceptos y calculando la distancia que existe entre el conocimiento del estudiante y el del sistema.

- El sistema indicará cuándo una respuesta es correcta o incorrecta y permitirá al estudiante indicar que no conoce la respuesta para una pregunta dada, realizada por el STI Iris.
- El sistema, además, permitirá al estudiante evadir una pregunta y posteriormente insistirá en que sea respondida.

Descripción interna de los módulos del STI Iris

En el **módulo pedagógico** del STI en cuestión residirá un agente pedagógico que posee “creencias”, “deseos” e “intenciones”, por lo cual podrá ser considerado además un agente BDI (*Beliefs-Desires-Intentions*: Creencias-Deseos-Intenciones). Este agente pedagógico estará a su vez conformado de varias piezas internas que le permitirán llevar a cabo sus funciones, entre ellas se encuentra un compilador (e intérprete) que permitirá analizar, compilar y ejecutar la notación intermedia que se generará para cada frase de entrada del usuario.

El **módulo de dominio** manejará la BC (conjunto de redes semánticas) y será dirigido directamente por el módulo pedagógico o agente pedagógico. Todas las acciones que se efectuarán sobre dicha BC serán desencadenadas por las decisiones tomadas en el agente pedagógico.

El **módulo de estudiante** manejará el perfil cognitivo de cada estudiante y también será dirigido por el agente pedagógico. Todas las acciones efectuadas sobre este módulo serán también desencadenadas por las decisiones tomadas en el agente pedagógico. Los perfiles cognitivos se modificarán con cada interacción de los estudiantes con el agente pedagógico.

2.2. Modelo de dominio

En este epígrafe se describe el negocio utilizando UML como lenguaje de modelado y AUP en su variante UCI como metodología de desarrollo de *software*, empleando uno de los artefactos que brinda: el modelo de dominio.

El modelo de dominio es utilizado por el analista como un medio para comprender el sistema, debido a que constituye una representación esquemática de los

conceptos y elementos de la vida real que serán usados en el mismo. Se crea con el fin de representar los conceptos claves del dominio del problema, las propiedades más importantes y las relaciones entre los conceptos, facilitando una mejor comunicación entre desarrolladores y clientes al establecer un lenguaje común para el entendimiento del mismo. (Eriksson & Penker, 2000)

El Diagrama de Clases del Modelo de Dominio (Figura 2.2) muestra una vista del modelo de dominio donde se pueden apreciar los elementos del problema modelado.

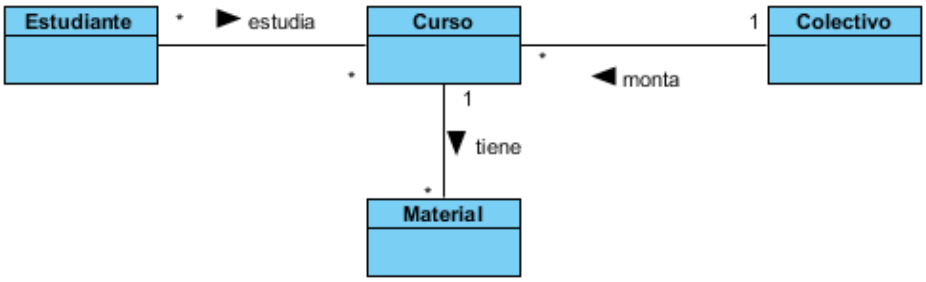

Figura 2.2. Diagrama de Clases del Modelo de Dominio.

Definición de las clases del modelo de dominio

Estudiante: El estudiante es la persona que se matricula a los cursos de la Plataforma educativa XAUCE ZERA, que consulta el contenido de sus cursos y que de forma opcional podrá consultar al STI para aclarar dudas y consolidar sus conocimientos.

Curso: Un curso montado en la Plataforma educativa XAUCE ZERA, que los estudiantes consultan y al cual están suscritos.

Colectivo: Un equipo multidisciplinario de profesores e ingenieros del conocimiento que montan el curso y entrenan al STI para que este, a su vez, pueda realizar sus funciones.

Material: Un material educativo montado en un curso, se refiere a documentos, libros, figuras y otros recursos como videos y audio.

2.3. Formato de definición de redes semánticas

Debido a la necesidad de incorporar redes semánticas al sistema que se propone, y de acuerdo a que estas deberán poseer características especiales en cuanto a la implementación prevista, se define en esta sección el formato que deberán cumplir para poder ser incluidas a la propuesta de solución.

Cada red semántica estará definida en un archivo con extensión ".json", cuyo contenido será precisamente un conjunto de datos definidos en JSON, de forma que al analizar dicho contenido, este contenga información válida, correctamente estructurada y respetando la sintaxis de dicho lenguaje de intercambio de datos.

El formato que se define a continuación consta de un conjunto de **reglas** que permiten definir cada nodo de la red a incluir, así como sus relaciones con otros nodos, relaciones especiales y otros aspectos.

Regla 1. El archivo contendrá un único objeto de JSON, que representa la red semántica que se define. La red semántica no podrá estar vacía, es decir, deberá definir al menos un nodo.

Regla 2. La definición de los nodos se especificará en el atributo "nodos" dentro del objeto único de red semántica. Este atributo consistirá en un arreglo de objetos, en los que cada uno definirá un nodo, este arreglo no podrá estar vacío.

Regla 3. Cada nodo tendrá los siguientes atributos obligatorios:

- Atributo "**nombre**": define el nombre del nodo. Debe ser único, para diferenciar un nodo de otro.
- Atributo "**tipo**": define el tipo del nodo. Cada nodo se clasificará como concepto o instancia, no ambos a la vez, esta clasificación se puede entender como analogía con el concepto de clase e instancia en la programación orientada a objetos (POO). Los posibles valores son: "concepto", "instancia".

Regla 4. Las relaciones entre los nodos de la red se definirán en el atributo opcional "relaciones" (un nodo puede o no tener relaciones con otros nodos). Este atributo

consistirá en un arreglo de objetos, en los que cada uno definirá una relación con otro nodo. Tradicionalmente, cada relación se define como una tripla de la forma “nodo origen – relación – nodo destino”, en este formato se declara el nodo origen como poseedor de las relaciones, por lo cual solo se define la relación del mismo con otros nodos. Cada relación puede tener cualquier nombre, exceptuando algunos nombres reservados por el formato, de acuerdo al tipo de nodo.

Regla 5. Cada relación se define como un objeto con un único atributo, que será el nombre de la relación y cuyo valor sea un arreglo de los nombres a los cuales apunta dicha relación.

Regla 6. Las relaciones reservadas por el formato y que forman parte de la lógica del tutor inteligente, están asociadas a cada tipo de nodo.

- Relaciones reservadas exclusivamente para los nodos de tipo “**concepto**”:
    - **super**: Define los nodos padres del nodo actual, de los cuales hereda todas sus propiedades (con o sin valor), consiste en un arreglo con los nombres de dichos nodos. Esto permite la herencia múltiple, es decir, que un nodo de tipo concepto pueda heredar propiedades o características de otros nodos de tipo concepto en la red semántica. Esta relación es **opcional** (un nodo concepto puede o no heredar de otro u otros nodos de tipo concepto).
- Relaciones reservadas exclusivamente para los nodos de tipo “**instancia**”:
    - **instancia de**: define cuál es el concepto del cual instancia el nodo actual. Esta relación es **obligatoria** en los nodos de tipo instancia (un nodo de tipo instancia puede y debe instanciar **uno y solo un** nodo de tipo concepto). Mediante la instanciación, el nodo actual hereda todas las propiedades (con o sin valor) del nodo de tipo concepto del cual instancia.

Regla 7. En el presente formato se define una propiedad como una relación de tipo “nodo origen – nombre – valor” donde:

- “nodo origen” representa el nodo actual, que posee la propiedad.

- “nombre” es el nombre de la propiedad.
- “valor” es el nodo destino, que servirá para almacenar el valor que toma la propiedad actual.

Regla 8. Las propiedades de un nodo se definen en el atributo “propiedades”, al mismo nivel que los demás atributos. El valor de este atributo será un arreglo donde cada elemento será un objeto propiedad. Cada objeto propiedad poseerá un único atributo, que será el nombre de la misma y cuyo valor podrá ser:

- Atómico: un único elemento o valor.
- Múltiple: un arreglo de elementos o valores.
- Sin valor: “null”, la propiedad tiene valor nulo o vacío.

Regla 9. Las propiedades son opcionales, es decir, un nodo puede o no poseer propiedades asociadas.

Regla 10. Si una propiedad está definida dentro de un nodo de tipo “instancia”, esta solo se aplica a dicho nodo. Si una propiedad está definida dentro de un nodo de tipo “concepto”, esta se aplica tanto al mismo como a todas sus instancias mediante un mecanismo de inferencia automática.

Regla 11. La propiedad “bibliografía” está reservada por el formato, es usada para definir el conjunto de fuentes bibliográficas asociadas a un nodo, definida mediante un arreglo o valor atómico.

El formato expuesto permite definir una red semántica en términos de conceptos, instancias y las propiedades de los mismos, tomando en cuenta sus relaciones.

2.4. Generación de respuestas y deducción de perfiles cognitivos

Para garantizar la generación de respuestas a cada frase de entrada del estudiante, así como la generación y deducción del perfil cognitivo de cada uno durante el proceso de comunicación humano-STI, se diseñó el algoritmo que se propone a continuación, en el cual son ejecutadas todas las operaciones de PLN y de IA.

I. El usuario envía una frase en lenguaje natural por medio de la GUI hacia la REST API del STI.

II. La REST API redirecciona la frase del usuario hacia el agente pedagógico (módulo pedagógico), el cual analiza y genera una respuesta, siguiendo estos pasos:

a. El agente pedagógico establece un contexto de ejecución que contiene la información útil como el usuario actual, el tema actual de conversación, el listado de las preguntas pendientes, las intenciones y otra información necesaria para responder la frase de entrada.

b. Se procesa la frase haciendo uso de las funcionalidades de la librería spaCy: Se hace pasar la frase del usuario a través de diferentes procesos, ejecutando operaciones de PLN como *tokenization*[3], el *tagging*[4], *parsing*[5], el reconocimiento de entidades nombradas, entre otras. Cuando el procesamiento de spaCy concluye, retorna un objeto que contiene la sentencia original y algunas capas de abstracción que proveen información acerca del mismo, este objeto es conocido como *doc*.

c. Se clasifica la frase de acuerdo a un conjunto de patrones sintácticos. Cuando la sentencia coincide con un patrón, es traducida a una notación intermedia, que será llamada *comando*. Este mecanismo permite al agente transformar la frase original del usuario a una notación que el agente pueda interpretar y ejecutar.

d. El compilador del agente pedagógico analiza léxica y sintácticamente el comando generado, posteriormente el intérprete lo ejecuta tomando en cuenta el contexto de ejecución generado en el paso *a*. En la implementación de cada comando reside el procesamiento de IA principal que define el comportamiento del agente pedagógico. Cuando un comando es ejecutado, retorna una respuesta que es generada dinámicamente.

[3] Proceso mediante el cual se divide la frase en elementos atómicos o pedazos llamados *tokens*.
[4] Asignación de etiquetas lingüísticas a cada token.
[5] Análisis sintáctico y construcción del árbol de dependencias sintácticas.

e. Usando los pedazos de respuesta generados en los pasos anteriores, el agente compone la respuesta final.

III. Se devuelve la respuesta final generada.

Durante este proceso de comunicación, el agente pedagógico dirige al módulo de estudiante para que este construya y actualice los perfiles cognitivos de los estudiantes. El módulo de estudiante es el responsable de actualizar las distancias que conforman la representación del modelo cognitivo del estudiante. En el presente trabajo se ha usado una medida de distancia definida en el rango de 0 a 1, donde 0 representa la total ignorancia acerca de un nodo y 1 representa el total conocimiento, esta distancia es calculada usando la siguiente fórmula:

$$d = \frac{1}{x}$$

En esta expresión *d* representa la distancia y *x* la cantidad de relaciones pendientes a chequear para un nodo específico, dadas por las intenciones del agente pedagógico. Cuando la última intención para un nodo determinado es llevada a cabo y evaluada, la distancia para ese nodo toma valor 1, lo cual representa que el nodo ha sido comprendido completamente de acuerdo al conocimiento almacenado en el sistema, posteriormente la intención es eliminada. Existe, por tanto, una relación inversa entre la distancia y la cantidad de relaciones restantes por chequear.

2.5. Especificación de requisitos

El levantamiento de los requisitos constituye una de las etapas más importantes en el proceso de ingeniería de requisitos. Para logar el éxito durante el desarrollo de esta etapa, se realiza un trabajo en conjunto entre los participantes y los desarrolladores con el objetivo de identificar el problema, proponer elementos de solución, negociar diferentes enfoques y especificar un conjunto preliminar de requisitos para la solución. (Pressman, 2010)

2.5.1. Requisitos funcionales

Los requisitos funcionales son declaraciones de los servicios que debe proporcionar el sistema, de la manera que éste debe reaccionar a entradas particulares y cómo se debe comportar en situaciones particulares (Sommerville, 2005). A continuación, se definen los requisitos funcionales del sistema:

RF1 Iniciar conversación

Permite al usuario autenticado en la Plataforma educativa XAUCE ZERA iniciar una conversación mediante la GUI del chat, enviando la información necesaria al servidor del STI.

RF2 Cerrar conversación

Permite al usuario autenticado en la Plataforma educativa XAUCE ZERA terminar una conversación mediante la GUI del chat, enviando la información necesaria al servidor del STI.

RF3 Enviar frase del usuario al STI

Permite enviar al servidor del STI una frase de entrada del usuario asociada a una conversación.

RF4 Procesar frase del usuario y enviar respuesta

Se procesa la frase en el servidor del STI mediante técnicas de IA, ejecutándose el comportamiento del STI para esa frase en particular, tomando en cuenta un contexto de ejecución y memoria de trabajo, con el objetivo de generar una grase de respuesta. Una vez finalizado el procesamiento y generada la frase, esta es enviada hacia la GUI del chat.

RF5 Guardar mensaje de conversación

Permite al STI guardar un mensaje en el historial de una conversación, asociada a un usuario.

RF6 Mostrar respuesta del STI a la frase del usuario

Muestra en el chat la frase de respuesta generada por el STI a la frase de entrada de usuario que fue previamente enviada y procesada.

RF7 Listar conversaciones archivadas

Permite al usuario autenticado en la Plataforma educativa XAUCE ZERA visualizar el listado de sus conversaciones previas archivadas.

RF8 Ver conversación archivada

Permite al usuario autenticado en la Plataforma educativa XAUCE ZERA visualizar una conversación previamente archivada.

RF9 Guardar información de ventana de chat en *cookie*

Guarda en una *cookie* en el navegador la información de la conversación actual y el estado de la GUI del chat, con el objetivo de mejorar la experiencia de usuario a través de la navegación en la Plataforma educativa XAUCE ZERA.

RF10 Cargar información de ventana de chat de *cookie*

Carga a partir de una *cookie* en el navegador la información de la conversación actual y el estado de la GUI del chat, con el objetivo de mejorar la experiencia de usuario a través de la navegación en la Plataforma educativa XAUCE ZERA.

RF11 Cargar/mostrar conversación anterior en el chat

Permite al usuario autenticado en la Plataforma educativa XAUCE ZERA visualizar la conversación anterior a la actual.

RF12 Minimizar ventana de chat

Permite al usuario autenticado en la Plataforma educativa XAUCE ZERA minimizar la ventana activa de chat.

RF13 Maximizar ventana de chat

Permite al usuario autenticado en la Plataforma educativa XAUCE ZERA maximizar la ventana activa de chat.

RF14 Cambiar opacidad de ventana de chat

Permite al usuario autenticado en la Plataforma educativa XAUCE ZERA cambiar la opacidad de la ventana de chat.

RF15 Ver chat en una página completa

Permite al usuario autenticado en la Plataforma educativa XAUCE ZERA visualizar el contenido del chat en una página con mayor espacio.

RF16 Incluir red semántica

Permite al administrador autenticado en la Plataforma educativa XAUCE ZERA incluir una red semántica asociada a un curso, a partir del archivo de definición de la misma, previamente elaborado.

RF17 Eliminar red semántica

Permite al administrador autenticado en la Plataforma educativa XAUCE ZERA eliminar una red semántica previamente incluida.

RF18 Listar redes semánticas

Permite al administrador autenticado en la Plataforma educativa XAUCE ZERA listar las redes semánticas incluidas en el sistema.

RF19 Buscar red semántica

Permite al administrador autenticado en la Plataforma educativa XAUCE ZERA buscar redes semánticas incluidas en el sistema.

RF20 Ver datos de red semántica

Permite al administrador autenticado en la Plataforma educativa XAUCE ZERA visualizar la representación gráfica de una red semántica incluida en el sistema.

RF21 Crear perfil cognitivo del estudiante

Permite al STI crear el perfil cognitivo del estudiante.

RF22 Listar perfiles cognitivos de los estudiantes

Permite al administrador autenticado en la Plataforma educativa XAUCE ZERA listar los perfiles cognitivos de los estudiantes previamente creados por el STI.

RF23 Ver perfil cognitivo del estudiante

Permite al administrador autenticado en la Plataforma educativa XAUCE ZERA ver los datos asociados a un perfil cognitivo.

RF24 Configurar conexión de XAUCE ZERA al STI

Permite al administrador autenticado en la Plataforma educativa XAUCE ZERA configurar la forma en que esta se conecta al servidor del STI para realizar peticiones.

RF25 Chequear manualmente conexión con servidor del STI

Permite al administrador autenticado en la Plataforma educativa XAUCE ZERA chequear que esta puede establecer conexión con el STI.

RF26 Chequear automáticamente conexión con servidor del STI

Permite a la Plataforma educativa XAUCE ZERA chequear automáticamente la conexión con el STI.

RF27 Personalizar identidad del STI

Permite al administrador autenticado en la Plataforma educativa XAUCE ZERA personalizar el nombre y la descripción del agente pedagógico que reside en el servidor del STI.

2.5.2. Requisitos no funcionales

Los requisitos no funcionales, son aquellos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las restricciones de los servicios o funciones ofrecidas por el sistema. Normalmente solo se aplican a características o servicios individuales del sistema (Sommerville, 2005). A continuación, se muestran los requisitos no funcionales definidos para la propuesta de solución:

Apariencia o interfaz externa

El diseño de la interfaz gráfica será minimalista, con pocas imágenes y colores, la información aparecerá correctamente organizada, permitiéndole al usuario encontrar con facilidad lo que desea. El color de los textos deberá contrastar con el fondo y el tamaño de fuente deberá ser lo suficientemente grande para que la información contenida sea fácil de leer. La interfaz gráfica reutilizará los conceptos de diseño asociados a la estrategia marcaria de la línea XAUCE de la UCI.

Usabilidad

El sistema podrá ser utilizado por cualquier usuario que tenga conocimientos básicos en el uso de aplicaciones web. Se realizará economía de clics, de forma que el usuario tenga que efectuar la menor cantidad posible para realizar una acción. El sistema debe notificar al usuario ante la presencia de un error y al concluir las acciones. Consistente: Siempre que sea posible las acciones similares deben ejecutarse de similar forma.

Rendimiento

Estará disponible las 24 horas del día y los 365 días del año. El tiempo de respuesta máximo estará en dependencia de la velocidad del servidor.

Portabilidad

La aplicación será multiplataforma.

Seguridad

Se garantiza la confidencialidad de los datos, así como su integridad. Se garantiza el acceso a las funcionalidades basado permisos por roles.

Requerimientos de *software*

Cliente: Debe tener instalado el navegador Firefox 57 o superior.

Servidor: El servidor del STI requerirá la instalación de las siguientes librerías de Python:

- SpaCy v2.0.7.
- Django v1.11.6.
- Django-cors-headers v2.2.0 o superior.
- Django rest framework v3.7.7 o superior.
- Matplotlib v2.2.2 o superior.
- Networkx v2.1 o superior.

2.6. Descripción de requisitos mediante Historias de Usuario

Rodríguez Sánchez plantea que la metodología seleccionada propone cuatro escenarios para modelar el sistema (Rodríguez Sánchez, 2015). En el presente trabajo se seleccionó el escenario no. 4, que es el modelado mediante Historias de Usuario (HU), esta selección se fundamenta en que el proyecto no modela negocio. La HU número 4 (Tabla 2.1) corresponde con el RF de igual número y nombre: Procesar frase del usuario y enviar respuesta.

Tabla 2.1. Ejemplo de HU

Número: 4	**Nombre del requisito:** Procesar frase del usuario y enviar respuesta
Programador: Carlos Ariel Rojas Lugones	**Iteración Asignada:** 2da
Prioridad: Alta	**Tiempo Estimado:** 1 mes
Riesgo en Desarrollo: Ninguno	**Tiempo Real:** 1 mes

Descripción:

Permite procesar la frase del usuario, llegar a un entendimiento de la misma en la medida de lo posible utilizando técnicas de Inteligencia Artificial y Procesamiento del Lenguaje Natural. Se determina respuesta y acción más adecuada para la frase de entrada.

1- Objetivo:

Realizar el procesamiento del lenguaje natural (frase del usuario) para determinar la respuesta y acción más adecuada de acuerdo a la red semántica por curso, teniendo en cuenta los nodos más importantes de la misma y el perfil cognitivo del estudiante.

2- Acciones para lograr el objetivo (precondiciones y datos):

Para poder procesar la frase del usuario es necesario:

El STI debe haber recibido la frase del usuario y el id de la conversación mediante una petición AJAX a la API usando el método POST a la URL “/api/conversation/sentence/send”.

3- Comportamientos válidos y no válidos (flujo central y alterno):

N/A

4- Flujo de la acción a realizar:

Los siguientes pasos del flujo ocurren totalmente del lado del STI:

- Se realiza el procesamiento del lenguaje natural a la frase de entrada del usuario. - Se traduce la frase a una notación intermedia. - Se ejecuta la notación intermedia, generando la frase de respuesta. - Se envía la frase de respuesta a la vista de XAUCE ZERA como parte de la respuesta de la solicitud AJAX. **5- Flujo alterno:** N/A
Observaciones: N/A
Prototipo elemental de interfaz gráfica de usuario: N/A

2.7. Arquitectura

La propuesta de solución consta de dos partes fundamentales:

- **STI**: Aplicación del *framework* Django, escrita en Python y que permitirá ser ejecutada mediante un servidor web. Prestará servicios a través de una API REST que será consumida por la plataforma. Será quien realizará el procesamiento en la propuesta de solución y consumirá datos de la plataforma a través de los servicios de la misma. Tendrá asociada su propia BD.
- **Plataforma educativa XAUCE ZERA**: Aplicación desarrollada con el *framework* Symfony, la cual tendrá un *bundle* en el cual estará el módulo de interfaz gráfica de usuario y que consumirá los servicios del STI mediante peticiones AJAX a la REST API del mismo. Tendrá asociada su propia BD.

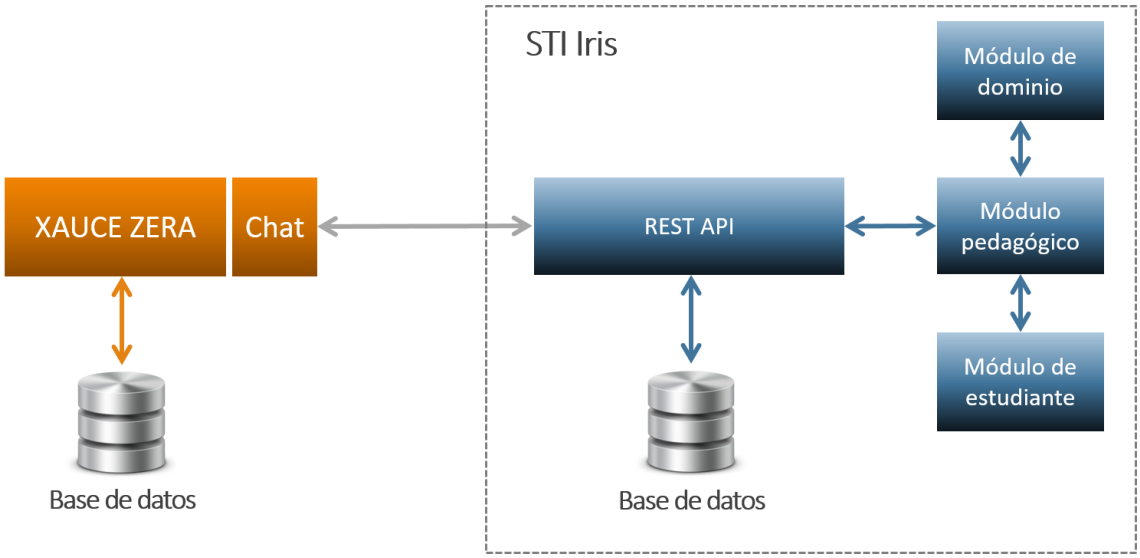

Figura 2.3. Arquitectura general de la propuesta de solución

Como se puede apreciar en la Figura 2.3 (izquierda), XAUCE ZERA, ubicada en un servidor con acceso a su propia BD, contiene la GUI del chat, el cual realiza peticiones al servidor del STI Iris (derecha). El STI Iris recibe las peticiones mediante la REST API, la cual desencadena el comportamiento definido en el módulo pedagógico, este módulo comanda directamente al módulo de dominio y al módulo de estudiante en el cumplimiento de sus responsabilidades. Es necesario enfatizar que la arquitectura propuesta en el STI cumple la arquitectura común de este tipo de sistemas, que presenta los módulos mostrados en la Figura 1.1.

2.7.1. Patrón arquitectónico

Los patrones arquitectónicos, o patrones de arquitectura, son patrones de diseño que ofrecen soluciones a problemas de arquitectura de software en ingeniería de software. Dan una descripción de los elementos y el tipo de relación que tienen junto con un conjunto de restricciones sobre cómo pueden ser usados. (Pressman, 2010)

Tanto el *bundle* de Symfony dentro de la plataforma como la aplicación de Django utilizan el patrón arquitectónico conocido como Modelo Vista Controlador (MVC: *Model View Controller*). A continuación, se describe cada componente de este patrón arquitectónico:

- **Modelo:** Es la capa que maneja los datos. Tiene comunicación únicamente con el controlador.
- **Vista:** Es la capa de presentación que ve el usuario. Es convocada y generada por el controlador y utiliza los datos que este le provee. La vista codifica y mantiene la presentación final de la aplicación, en ella reside el código HTML, CSS y JavaScript (lenguajes del lado del cliente) que se tiene que generar para producir la página que verá el usuario.
- **Controlador:** La capa lógica que realiza el procesamiento de las peticiones, maneja el modelo y genera vistas que son devueltas al cliente.

Adicionalmente, cabe destacar que, en el caso de la aplicación escrita en Django, esta utiliza una variación del patrón MVC conocida como Modelo Vista Template (MVT), que es una adaptación realizada a dicho patrón por parte de la comunidad de desarrollo del *framework* Django. Esta adaptación plantea que los nombres estándares “modelo”, “vista” y “controlador” son debatibles y defiende la postura de que la vista describa qué datos son mostrados y no cómo son mostrados. Más adelante, este modelo separa el contenido de su forma de presentación, de lo cual se encargan los *templates* (plantillas). El controlador queda disuelto en el entramado del *framework*, es decir, el *framework* se convierte en el controlador: la maquinaria que envía una petición a la vista, de acuerdo a la configuración de las URL de Django. (Django Documentation, 2018)

2.8. Modelo de diseño

El modelo de diseño se crea tomando como base el modelo de análisis para modelar el sistema convirtiéndose en un punto de partida para la implementación. (Jacobson & James, 2000)

2.8.1. Patrones de diseño

Larman en su libro “UML y Patrones” plantea que los desarrolladores acumulan un repertorio tanto de principios generales como de soluciones basadas en aplicar ciertos estilos que les guían en la creación de *software*. Estos principios y estilos, codificados adecuadamente, describiendo el problema y la solución son conocidos

como patrones. Un patrón es un par problema/solución con nombre que se puede aplicar en nuevos contextos, con consejos acerca de cómo aplicarlo en dichas situaciones y discusiones sobre sus compromisos. (Larman, 2003)

Los GRASP (*General Responsibility Assignment Software Patterns* o patrones generales de *software* para asignar responsabilidades) son principios utilizados en el diseño exitoso de *software* orientado a objetos. (Larman, 2003)

Para la implementación de la propuesta de solución, se utilizarán los siguientes patrones GRASP descritos en (Larman, 2003):

- **Experto en Información (o Experto)**: Plantea que se debe asignar una responsabilidad al experto en información, es decir, la clase que tiene la información necesaria para realizar la responsabilidad.
- **Creador**: Asignar a una clase la responsabilidad de crear una instancia de otra clase si se cumplen determinadas condiciones.
- **Bajo acoplamiento**: Asignar una responsabilidad de manera que el acoplamiento[6] permanezca bajo.
- **Alta cohesión**: Asignar una responsabilidad de manera que la cohesión[7] permanezca alta.
- **Controlador**: Asignar la responsabilidad de recibir o manejar un mensaje de evento del sistema a una clase.

En el libro *Design Patterns* se presentan 23 patrones que son útiles durante el diseño de objetos. Debido a que el libro fue escrito por cuatro autores, estos patrones se conocen como los patrones de la “pantilla de los cuatro” o patrones “GoF”. (Larman, 2003)

[6] El acoplamiento es una medida de la fuerza con que un elemento está conectado a, tiene conocimiento de, confía en, otros elementos (Larman, 2003).

[7] La cohesión funcional es una medida de la fuerza con la que se relacionan y del grado de focalización de las responsabilidades de un elemento. Un elemento con responsabilidades altamente relacionadas, y que no hace una gran cantidad de trabajo, tiene alta cohesión.

En la propuesta de solución se utilizarán los siguientes patrones GoF tomando en cuenta la descripción dada por los autores del libro *Design Patterns* (Gamma, Helm, Johnson, & Vlissides, 2009):

- **Singleton:** Se asegura que una clase tenga una única instancia, y se provee un punto de acceso global para la misma.
- **Decorator:** Se le asignan responsabilidades de forma dinámica a un objeto. Los decoradores proveen una alternativa flexible a la herencia para extender las funcionalidades.
- **Proxy:** Provee un objeto de control de acceso para otro objeto.
- **Command:** Encapsula una petición como un objeto, permite la ejecución de acciones.
- **Interpreter:** Dado un lenguaje, define la representación de su gramática, así como un intérprete que la utiliza para ejecutar las sentencias del lenguaje.

2.8.2. Diagrama de clases del diseño

Un diagrama de clases del diseño (DCD) representa las especificaciones de las clases e interfaces *software* (por ejemplo, las interfaces de Java) en una aplicación. Entre la información general se encuentra (Larman, 2003):

- Clases, asociaciones y atributos.
- Interfaces, con sus operaciones y constantes.
- Métodos.
- Información acerca del tipo de atributos.
- Navegabilidad.
- Dependencias.

A diferencia de las clases del Modelo de Dominio, las clases de un DCD muestran las definiciones de las clases del *software* en lugar de los conceptos del mundo real (Larman, 2003). En la Figura 2.4, se muestra el DCD de la propuesta de solución, donde se aprecian los paquetes y las clases que definen dicho módulo.

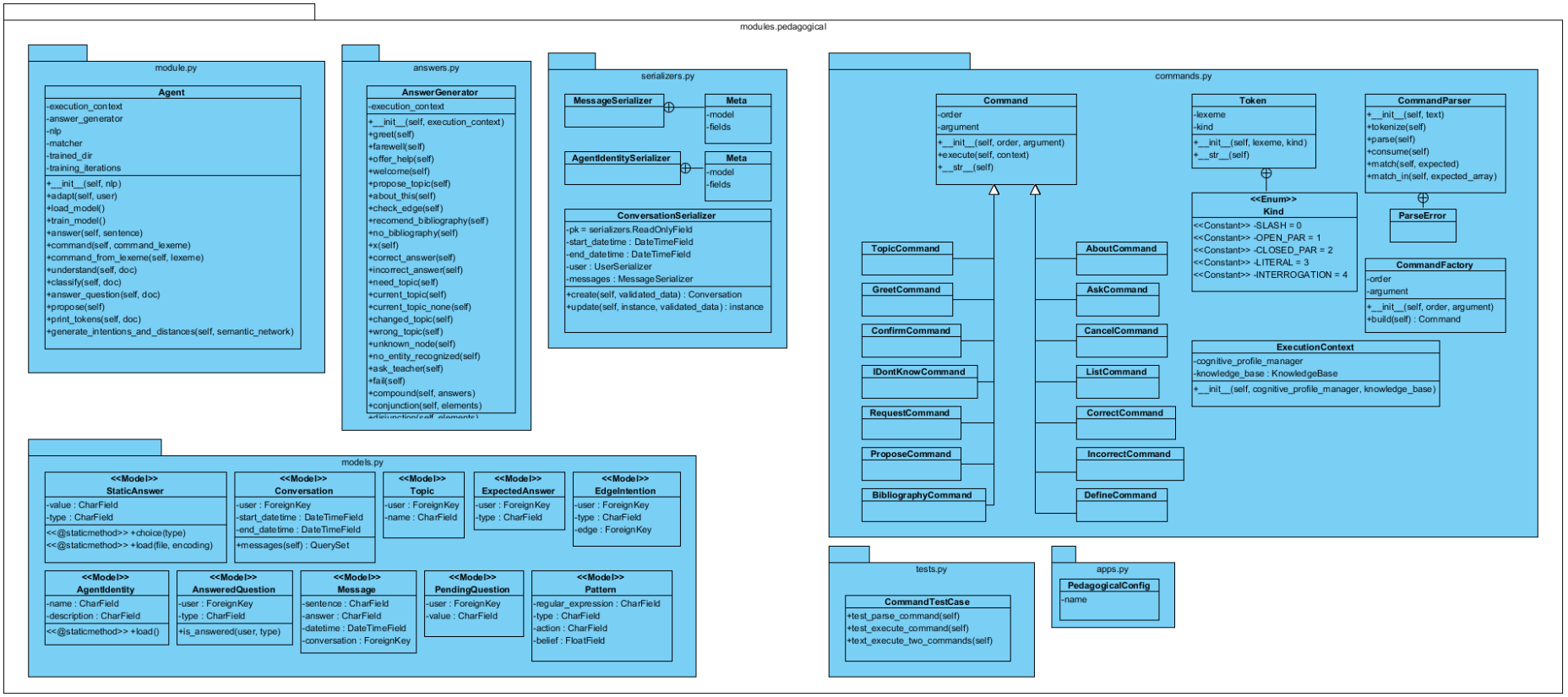

Figura 2.4. Diagrama de Clases del Diseño correspondiente al módulo pedagógico

2.9. Modelo de despliegue

Diagrama de despliegue

UML define varios diagramas que se pueden utilizar para ilustrar los detalles de implementación. Con el objetivo de ilustrar el despliegue de los componentes y procesos en los nodos de proceso, se utiliza el diagrama de despliegue. (Larman, 2003)

La Figura 2.5 muestra el diagrama de despliegue de la propuesta de solución.

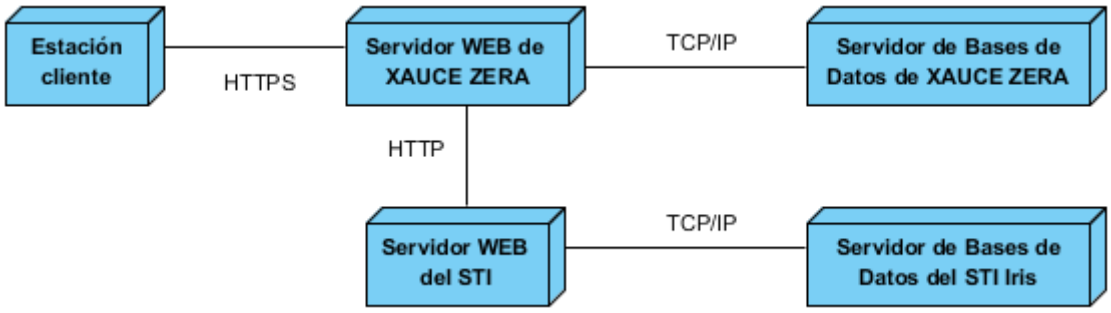

Figura 2.5. Diagrama de despliegue.

En la solución propuesta el cliente accede al sistema usando el protocolo HTTPS a través de un navegador. El sistema está dividido en dos servidores web, uno que contiene a la Plataforma educativa XAUCE ZERA y otro que contiene al STI, la plataforma realiza las peticiones al STI usando AJAX y ambos se conectan a su propio servidor de bases de datos utilizando el protocolo TCP/IP.

2.10. Diseño de la Base de Datos

La BD de la Plataforma educativa XAUCE ZERA consta de un conjunto de paquetes o esquemas que contienen una serie de tablas por cada uno, dichas tablas corresponden a cada una de las funcionalidades con las que cuenta la plataforma. En el caso de la propuesta de solución se plantea una nueva BD, independiente de la BD de la plataforma, que contendrá la información que necesitará persistir el STI.

2.10.1. Diagrama entidad-relación

El diagrama entidad-relación es uno de los modelos más usados para diseñar bases de datos, este modelo se encuentra basado en dos conceptos fundamentales: entidades, que representan objetos sobre los cuales se desea

guardar información y las relaciones, que constituyen las relaciones entre las entidades. (Ruiz, 2000)

El diagrama entidad-relación de la propuesta de solución (Figura 2.6) se divide en dos paquetes: el superior, que contiene las tablas propias del STI y el inferior que contiene las tablas que almacenan los datos importados desde la Plataforma educativa XAUCE ZERA.

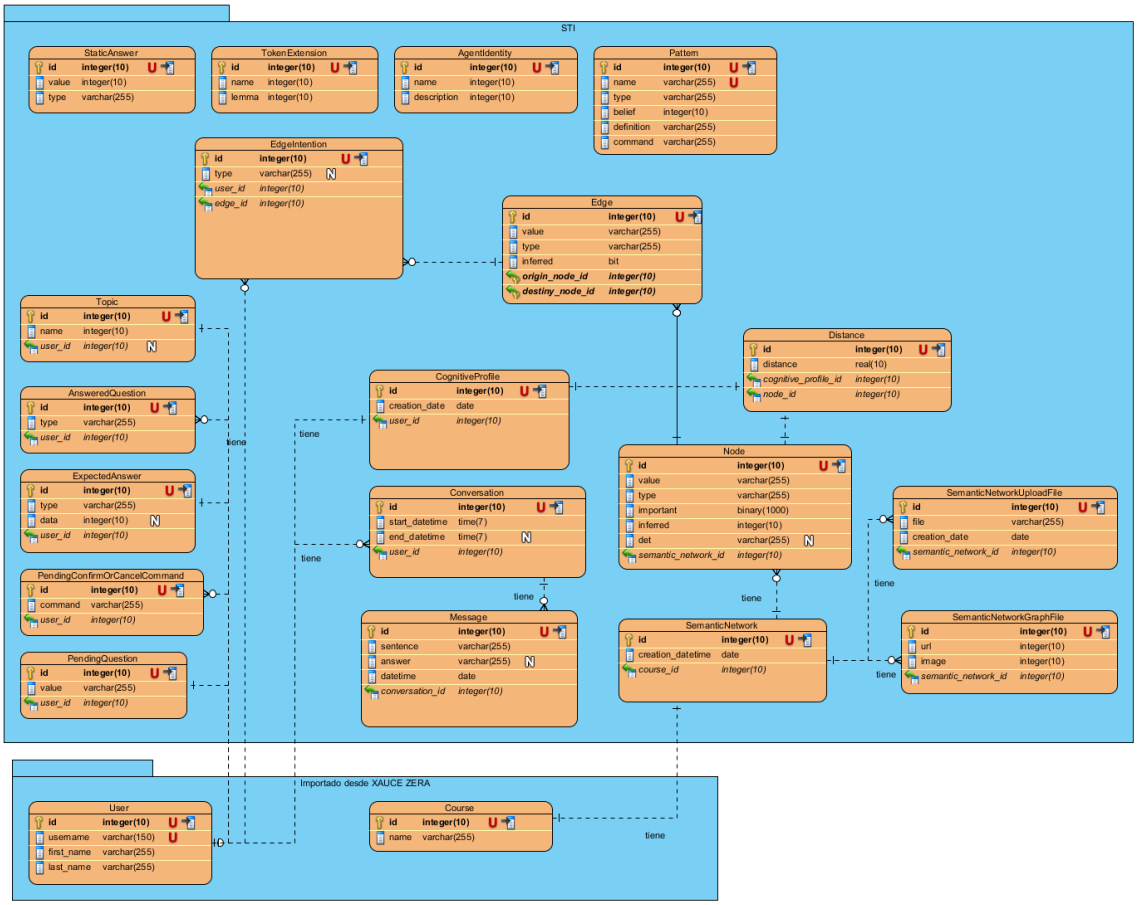

Figura 2.6. Diagrama entidad-relación

2.10.2. Descripción de las tablas de la base de datos

A continuación, se describen las tablas de la BD propuesta del lado del STI.

Course: Almacena la información importada desde la plataforma relacionada con cada curso de la misma.

User: Almacena la información importada desde la plataforma relacionada con cada usuario de la misma.

SemanticNetwork: Define una red semántica, asociada con un curso.

Node: Define un nodo asociado a una red semántica y tiene información asociada.

Edge: Relaciona dos nodos y tiene información asociada.

SemanticNetworkGraphFile: Relaciona una red semántica con una imagen PNG que es creada y que representa gráficamente dicha red.

SemanticNetworkUploadFile: Relaciona una red semántica con su archivo original de definición.

Topic: Asocia a cada usuario un tema actual de conversación.

Pattern: Lista los patrones lingüísticos que al encontrarse en la frase original del usuario desencadenan un procesamiento.

AnsweredQuestion: Mantiene una lista que permite recordar las preguntas que ha respondido a cada usuario.

StaticAnswer: Almacena las respuestas estáticas que sirven de base en la construcción de respuestas dinámicas.

ExpectedAnswer: Asocia a cada usuario una respuesta esperada lo cual se usa para intentar predecir las frases del usuario.

PendingQuestion: Mantiene una lista que permite recordar las preguntas pendientes que se han realizado al usuario y que no ha respondido.

TokenExtension: Almacena extensiones de la clase Token de la librería spaCy.

CognitiveProfile: Define el perfil cognitivo del estudiante, es decir, el seguimiento y deducción de distancias que realiza el STI sobre el estudiante.

Distance: Define las distancias deducidas por el STI para cada estudiante en cada nodo importante de la red semántica.

Conversation: Define la conversación establecida entre el estudiante y el STI.

Message: Permite persistir la frase enviada por el usuario y la respuesta dada por el STI, así como la fecha y hora en que ocurren.

EdgeIntention: Forma parte de la memoria de trabajo del STI, registra las intenciones del mismo con respecto a cada estudiante. Establece por estudiante,

para cada arista de la red semántica asociada a cada nodo importante, qué acciones se deben realizar. El campo tipo almacena la acción, que puede ser: aclarar duda o chequear objetivo.

AgentIdentity: Almacena la identidad del agente o tutor inteligente, la cual es personalizable.

2.11. Conclusiones parciales

La ejecución de la disciplina “Análisis y diseño”, definida en la metodología seleccionada, permitió determinar las características fundamentales de la propuesta de solución, con lo cual se sentaron las bases para la posterior implementación del sistema.

El análisis llevado a cabo permitió realizar un diseño inicial de las clases, paquetes y módulos del sistema, las cuales estarán en constante perfeccionamiento en la fase de implementación.

Capítulo 3. Implementación y prueba de la propuesta de solución

El presente capítulo aborda las disciplinas de Implementación, Pruebas Internas y Pruebas de Aceptación de la solución propuesta, tal y como propone la metodología de desarrollo de *software* AUP en su variante UCI.

3.1. Implementación

Una vez realizadas las actividades asociadas a la disciplina de Análisis y Diseño, y partiendo de los resultados obtenidos, se procede a la implementación del sistema, tal y como indican los objetivos de la disciplina de Implementación de la metodología seleccionada (Rodríguez Sánchez, 2015).

3.1.1. Generación de respuestas

El STI implementado es capaz de generar respuestas de forma dinámica para cada frase de entrada del estudiante. La generación de estas respuestas se realiza a partir de pedazos estáticos que son generados en etapas intermedias y posteriormente unidos para componer la respuesta final.

Al iniciarse el servidor por primera vez, el sistema lee desde un archivo llamado “static_answers.json” (Figura 3.1) el conjunto de respuestas estáticas que utilizará para componer las frases finales. Cada frase estática leída es posteriormente escrita en la BD con el objetivo de mejorar el tiempo de acceso, evitando leer el archivo una y otra vez durante la ejecución del sistema.

El archivo mencionado deberá cumplir con un formato sencillo en el cual el objeto “static_answers” tendrá varios atributos o campos, los cuales serán los tipos de respuestas estáticas definidas, y a su vez, estos atributos tendrán como valor un arreglo de las posibles respuestas.

Con el objetivo humanizar la conversación, el sistema selecciona las respuestas estáticas al azar, correspondiéndose con un tipo específico de respuesta estática.

```
{
  "static_answers": {
    "WELCOME": [
      "Hola, te doy la bienvenida",
      "¡Saludos! Te doy la bienvenida",
      "Un cordial saludo, te doy la bienvenida",
      "Hola, es la primera vez que hablamos, te doy la bienvenida"
    ],
    "GREETING": [
      "Hola",
      "Buenas",
      "Saludos",
      "¡Hey hola!",
    ]
  }
}
```

Figura 3.1. Ejemplo de respuestas estáticas definidas en JSON.

En el archivo de frases estáticas mencionado se definieron 27 tipos de respuestas estáticas, conteniendo un conjunto asociado a cada una, para un total de 300 frases.

Se definió además un algoritmo para realizar la conjunción y disyunción de elementos, permitiendo al STI Iris ser capaz de listar elementos unidos por conjunciones o disyunciones. Para el algoritmo definido, dado un conjunto ejemplo $\{ A, B, C \}$, el sistema genera como resultado de la conjunción la frase “A, B y C”, y para la disyunción la frase “A, B o C”. Así mismo, el sistema toma en cuenta las reglas del idioma español y utiliza la conjunción “e” en lugar de “y” si la siguiente palabra comienza en “i”, igualmente en el caso de la disyunción utiliza “u” en lugar de “o” si la siguiente palabra comienza en “o”. La conjunción o disyunción de conjuntos de un solo elemento generarán como resultado el propio elemento.

Dentro del agente pedagógico existe un objeto generador de respuestas que se encarga de seleccionar la frase adecuada según el tipo de respuesta que se debe generar. Este objeto es el responsable de componer la frase final, tomando en cuenta los elementos y algoritmos anteriormente mencionados.

3.1.2. Manejo de perfiles cognitivos

El agente pedagógico dirige al módulo de estudiante, desencadenando un conjunto de acciones que permiten crear y actualizar los perfiles cognitivos de los estudiantes. Cuando el estudiante se comunica por primera vez con el STI Iris, este crea un perfil cognitivo del mismo, que genera las distancias para cada

nodo importante de cada red semántica, inicializándolas en cero. Cuando el estudiante acierta una pregunta realizada por el sistema, este recalcula y ajusta la distancia para el contenido o nodo asociado a dicha pregunta.

3.1.3. Notación intermedia basada en comandos

Se implementó un total de 15 comandos (Tabla 3.1), los cuales definen el comportamiento de IA del sistema y constituyen el núcleo de la lógica del agente inteligente.

Tabla 3.1. Ejemplo de comandos implementados

Sintaxis	Descripción
/greet()	Saluda o da la bienvenida al usuario.
/farewell()	Se despide del usuario y cierra la conversación actual.
/about(this)	Se identifica, ofreciendo información sobre sí mismo.
/topic(argument)	Cambia o menciona el tema de conversación actual.
/define(argument)	Define un concepto a partir de la BC.
/bibliography(argument)	Recomienda bibliografía para un contenido de la BC.
/ask(content)	Realiza una pregunta de control al estudiante.

Uno de los comandos más complejos es "*/topic(argument)*", que indica al agente definir el concepto (o argumento) contenido entre paréntesis. Su implementación consiste en que el agente pide a la BC el nodo a definir, se mueve a los nodos adyacentes al mismo y genera el concepto basándose en las relaciones en común con otros nodos. Si el algoritmo no encuentra relaciones directas, entonces efectúa el proceso a la inversa, partiendo de otros nodos y llegando al nodo objetivo, logrando aportar la mayor información posible al estudiante. Cabe destacar que la definición de un concepto no se genera a partir de una respuesta estática, sino que es construido dinámicamente. De forma similar se procede al recomendar fuentes bibliográficas.

Otro comando que posee una implementación compleja es "*/ask(content)*", que indica al agente generar una pregunta de control al estudiante, tomando en cuenta las relaciones de un contenido específico y teniendo en cuenta el perfil

cognitivo. La implementación de este comando toma en cuenta si una relación ha sido o no comprobada con anterioridad.

3.1.4. Clasificación de frases utilizando patrones sintácticos

El proceso que atraviesa una frase de entrada del estudiante para ser traducida a una notación intermedia introduce la necesidad de clasificarla de acuerdo a un conjunto de patrones sintácticos. Se definió en el archivo “patterns.json” el conjunto de patrones que maneja el STI Iris, para un total de 13. El contenido de este archivo es cargado al sistema de forma análoga al proceso que maneja el archivo de respuestas estáticas.

Cada patrón sintáctico (Figura 3.2) definido en el archivo, contiene la siguiente información: el nombre único del patrón, el cuerpo de su definición (una secuencia de *tokens*), su tipo y el comando asociado al mismo.

```
{
  "name": "DEFINE",
  "definition": [
    [{"LEMMA": "definir"}, {"POS": "NOUN"}],
    [{"LEMMA": "definir"}, {"POS": "PROPN"}]
  ],
  "type": "order",
  "command": "/define(x)"
}
```

Figura 3.2. Ejemplo de patrón sintáctico definido en JSON.

La Figura 3.2 muestra un patrón llamado “DEFINE” que es activado cuando la frase de entrada coincide parcialmente con las siguientes secuencias de palabras (*tokens*):

- Palabra cuyo lema sea “definir” + sustantivo común.
- Palabra cuyo lema sea “definir” + sustantivo propio.

El patrón descrito es una orden del estudiante y se traduce a la notación intermedia o comando “*/define(x)*” donde “x” es el nombre de la entidad encontrada, que debe coincidir con el nombre de un nodo de la red semántica que representa el tema actual de conversación. Un ejemplo de frase que coincide con este patrón es “define agente”, la cual es interpretada como “*/define(agente)*” y arroja como resultado la definición de dicho concepto.

3.1.5. Procesamiento del archivo de definición de red semántica

Un ejemplo de cómo codificar una red semántica utilizando el formato definido en la sección 2.3 es la definición del concepto “Inteligencia Artificial” (Figura 3.3). En la parte izquierda de la figura se muestra el código fuente del archivo JSON que define la red mediante el formato definido. La parte derecha de la figura muestra la representación gráfica del nodo principal y los nodos generados automáticamente al ser procesado el archivo de red semántica.

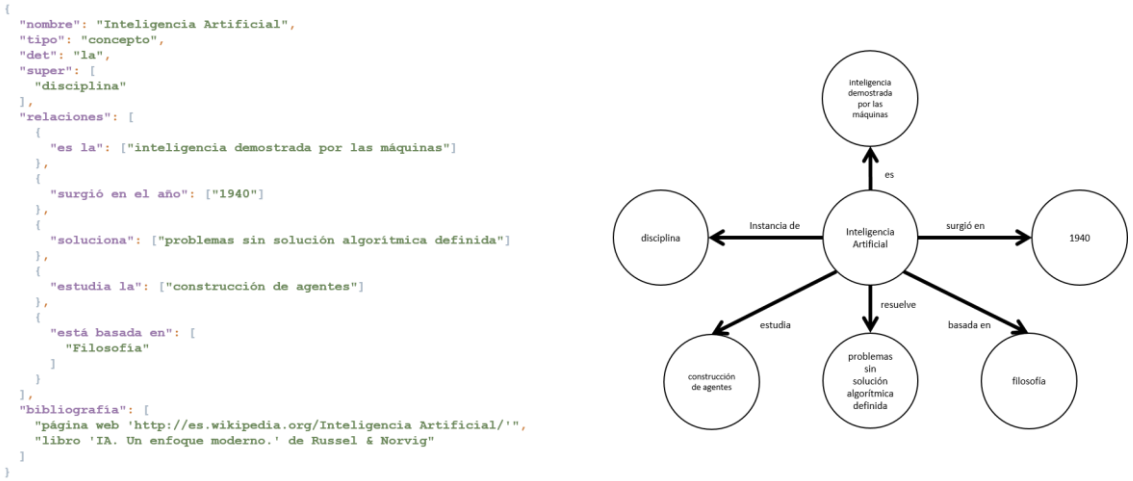

Figura 3.3. Ejemplo de definición de red semántica y su representación gráfica.

Toda la información definida en cada archivo de definición de red semántica incorporada al sistema es transformada y posteriormente almacenada en la BD relacional, mejorando el tiempo de acceso y permitiendo al agente pedagógico moverse sobre la red semántica con mayor facilidad para realizar inferencias y definiciones de conceptos.

3.1.6. Manejo de intenciones en el agente pedagógico

Las intenciones pueden definirse como acciones que el agente pedagógico tiene pendiente a realizar. Se implementó el sistema de intenciones del STI Iris de forma que cuando se añade una nueva red semántica, se registra una intención por cada nodo importante de la misma, asociada a cada usuario. Estas intenciones son posteriormente utilizadas en la generación de preguntas de control, debido a que dichas preguntas están ligadas a las intenciones. Una vez que el sistema completa y elimina las intenciones, no realiza más preguntas de control al estudiante.

3.1.7. Manejo de preguntas pendientes y respuestas esperadas

Cuando el STI Iris realiza una pregunta de control al estudiante, intenta predecir la posible respuesta del mismo utilizando un sistema de respuestas esperadas. Si la respuesta del estudiante es subconjunto del conjunto de respuestas esperadas, entonces se establece la respuesta como correcta, en caso contrario se marca como incorrecta y no se elimina la intención que generó la pregunta.

El sistema fue implementado de forma tal que, si el estudiante no responde a la pregunta del STI Iris, este la registra como pregunta pendiente y sigue preguntándola hasta que el estudiante responda correctamente o declare su desconocimiento del contenido en particular.

3.1.8. Librerías utilizadas

El PLN y otras actividades de alta complejidad demandan el uso de herramientas maduras y flexibles con una probada eficiencia. En la familia de librerías de PLN de Python se encuentra **spaCy**, una librería “de fuerza y potencia industrial para el PLN” (Sitio oficial de spaCy, 2018).

La selección de esta librería está fundamentada en diferentes referencias: La investigación de Choi y colectivo de autores en 2015, mostró que spaCy “ofrece el analizador sintáctico más rápido del mundo y su exactitud estuvo en el 1% de los mejores disponibles” (Choi, Tetreault, & Stent, 2015). La versión actual, 2.0, utilizada en el presente trabajo, según señalan los desarrolladores “es más exacta que cualquiera de los sistemas que Choi y colectivo de autores evaluó”, otra de las razones que fundamentan la elección de esta librería es que la misma ha servido de soporte para nuevos proyectos, librerías y tecnologías, por ejemplo Torchtext y AllenNLP (Sitio oficial de spaCy, 2018).

Matplotlib (2.2.2) es una librería de trazado en 2D para Python que produce figuras a calidad de publicación en una variedad de formatos de copia impresa y de entornos interactivos a través de diferentes plataformas. Esta librería puede ser usada en los *scripts* de Python, así como en servidores de aplicaciones web, así como *toolkits* de GUI. Permite generar histogramas, gráficos espectrales, gráficos de barras, gráficos de error, entre otros, mediante muy pocas líneas de código fuente. (Matplotlib official site, 2018)

Networkx (2.1) es una librería de Python para la creación, manipulación y estudio de la estructura, dinámica y funciones de redes complejas. Algunas de sus funcionalidades son el manejo de estructuras de datos para grafos, dígrafos y multigrafos, el soporte de diferentes algoritmos de teoría de grafos, la generación de grafos clásicos, grafos aleatorios y redes simétricas. Es de código abierto y ha sido ampliamente probada, además incluye funcionalidades de prototipado rápido y es multiplataforma. (Networkx, 2018)

Django-cors-headers es una aplicación o módulo reutilizable del *framework* Django que añade cabeceras CORS (*Cross-Origin Resource Sharing*) a las respuestas HTTP del servidor, lo cual permite que al mismo se le puedan realizar peticiones AJAX entre diferentes dominios (*cross-domain requests*) de forma similar a como se realiza en peticiones que provienen del mismo dominio. (Documentación de Django-cors-headers, 2018)

3.1.9. Diagrama de componentes

Los diagramas de componentes se utilizan para modelar la vista estática de un sistema. Comprende entre sus principales objetivos mostrar la dependencia entre los distintos componentes del *software* y representar las relaciones entre los elementos que forman el código del sistema implementado. (Jacobson & James, 2000)

El diagrama de componentes correspondiente al servidor del STI (Figura 3.4) muestra los componentes que conforman al servidor del STI Iris, el cual se divide en el *kernel*, la REST API y los módulos. El *kernel* es el núcleo del proyecto, el cual se encarga de cargar la configuración del servidor, así como de recibir las peticiones y desviarlas a las otras capas del sistema.

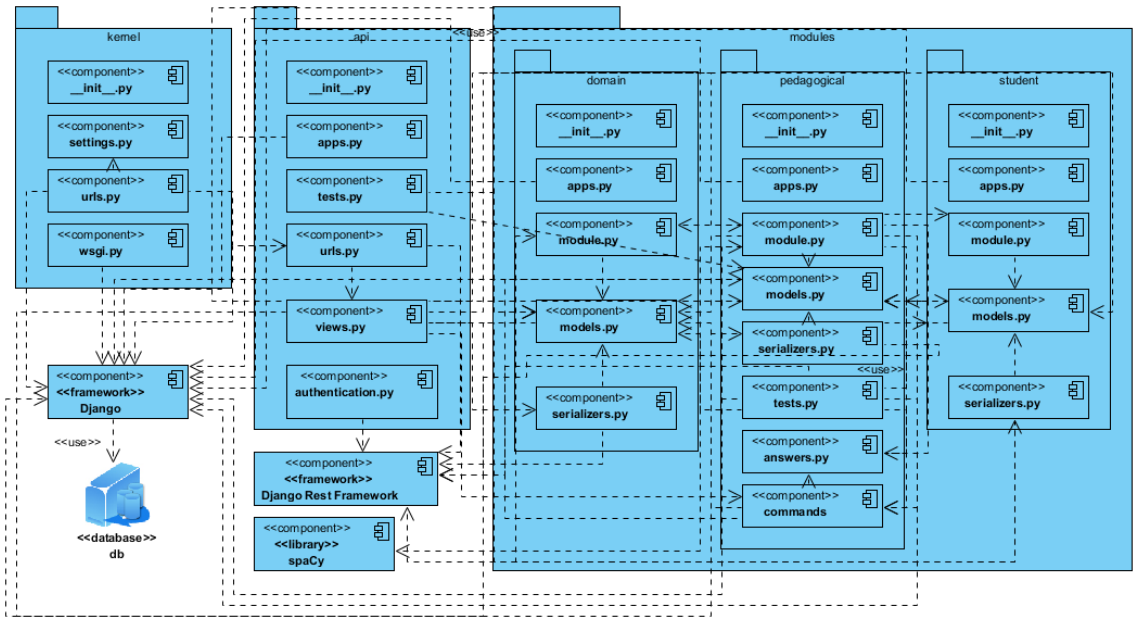

Figura 3.4. Diagrama de componentes del servidor del STI Iris.

Como se aprecia en el diagrama de componentes correspondiente al *bundle* que integra la propuesta de solución con la Plataforma educativa XAUCE ZERA (Figura 3.5), dicho *bundle* se divide en los paquetes y componentes característicos de Symfony y la arquitectura de referencia del Centro FORTES, llamada XALIX.

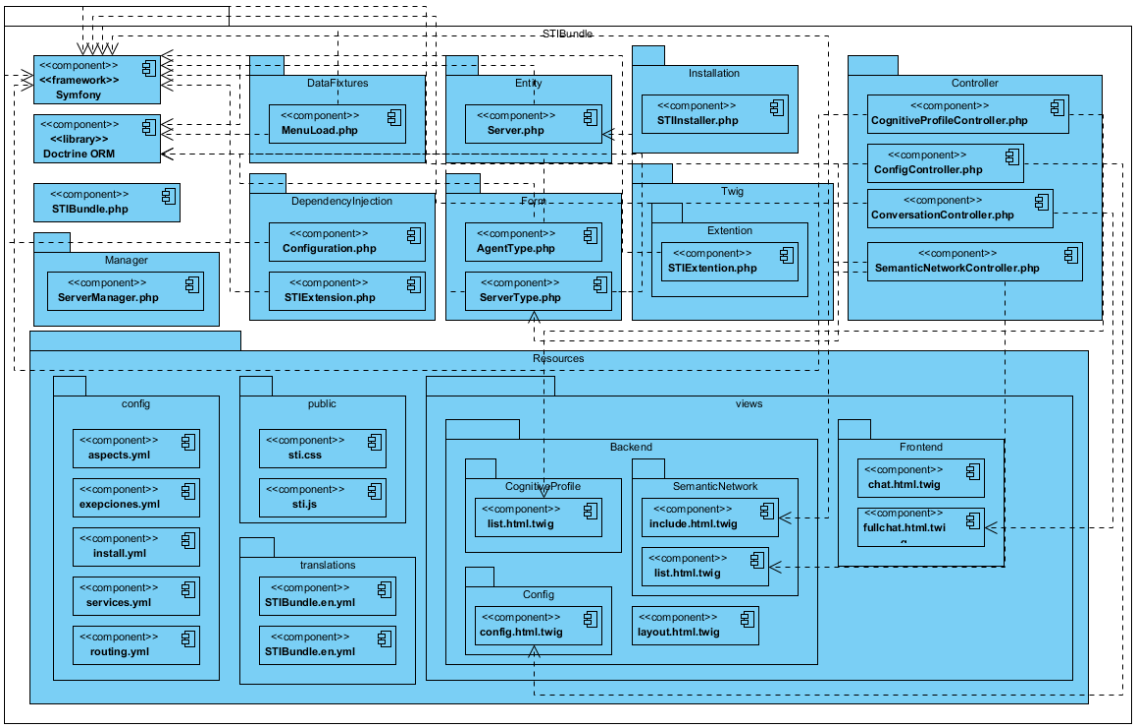

Figura 3.5. Diagrama de componentes de STIBundle en XAUCE ZERA

3.2. Pruebas internas

La disciplina de Pruebas internas verifica el resultado de la implementación, probando cada construcción, incluyendo tanto las construcciones internas como intermedias. Durante la ejecución de las actividades asociadas a esta disciplina, se generan los artefactos necesarios, entre ellos los Diseños de casos de prueba (Rodríguez Sánchez, 2015).

3.2.1. Pruebas unitarias

En las metodologías tradicionales, la fase de pruebas, incluyendo la definición de los *tests*, es usualmente realizada sobre el final del proyecto, o sobre el final del desarrollo de cada módulo. Otras metodologías como XP proponen un modelo inverso en el que lo primero que se escribe son los *tests* que el sistema debe pasar. Luego, el desarrollo debe ser el mínimo para pasar las pruebas previamente definidas. Estas pruebas son llamadas **pruebas unitarias**, las cuales son escritas y realizadas por los programadores. La definición de estos *tests* al comienzo, condiciona o dirige el desarrollo, esta forma de trabajo es conocida como "*test-driven development*" (desarrollo dirigido por pruebas). (Joskowicz, 2008)

En la implementación del servidor del STI Iris se siguió el enfoque de desarrollo dirigido por pruebas, debido a que la comunidad del *framework* Django exhorta esta práctica de acuerdo a sus múltiples ventajas: "Los programadores escriben sus pruebas antes que su código, esto puede parecer contra-intuitivo, pero de hecho es similar a cómo las personas resuelven sus problemas, primero describen el problema y luego crean código para solucionarlo. El desarrollo dirigido por pruebas formaliza el problema en un caso de prueba escrito en Python." (Django Documentation, 2018).

Las pruebas unitarias realizadas sobre la REST API del STI, fueron llevadas a cabo utilizando dos métodos distintos, para asegurar un mejor resultado en las pruebas: Se escribieron pruebas unitarias en Python, usando la ***suite* de pruebas de Django**, llamada ***unittest***, la cual permite ejecutar con facilidad las pruebas escritas y depurar posibles no conformidades o resultados inesperados. Además, se utilizó **Postman**, una herramienta líder especialmente diseñada para probar servicios *restful*, utilizada por los desarrolladores para desarrollar

software moderno que siga el enfoque *API-first* (desarrollar la REST API primero) (Sitio oficial de Postman, 2018).

La Figura 3.6 muestra los resultados arrojados en las diferentes iteraciones de pruebas unitarias realizadas sobre la REST API del STI. En cada iteración se puede observar (en verde) cuáles fueron las pruebas que pasaron, en amarillo las que arrojaron advertencias y en rojo las que no pasaron. Las tres iteraciones realizadas muestran que a lo largo del proceso de desarrollo fueron solucionándose los errores detectados, realizando las modificaciones pertinentes al código fuente para eliminar las advertencias y las funcionalidades requeridas.

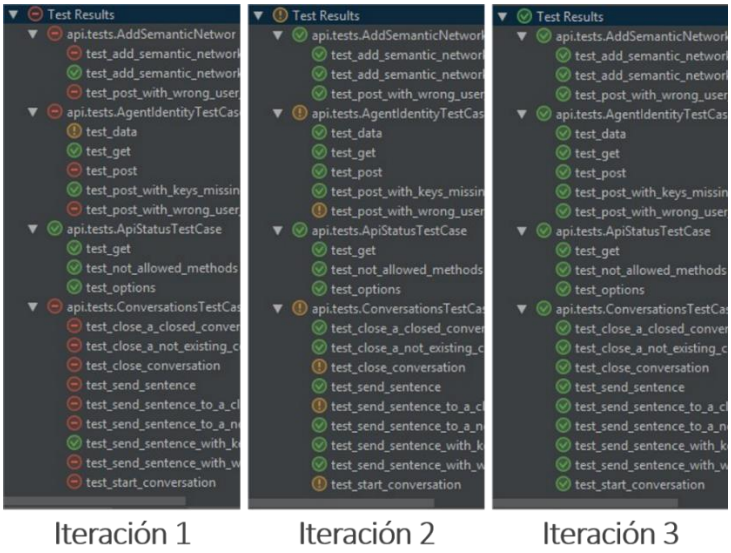

Figura 3.6. Capturas de pantalla de los resultados de las pruebas unitarias utilizando *django test suite*.

Estas pruebas arrojaron como resultados (Figura 3.7) una advertencia y siete errores en la primera iteración, cuatro advertencias en la segunda y ningún error o advertencia en la tercera.

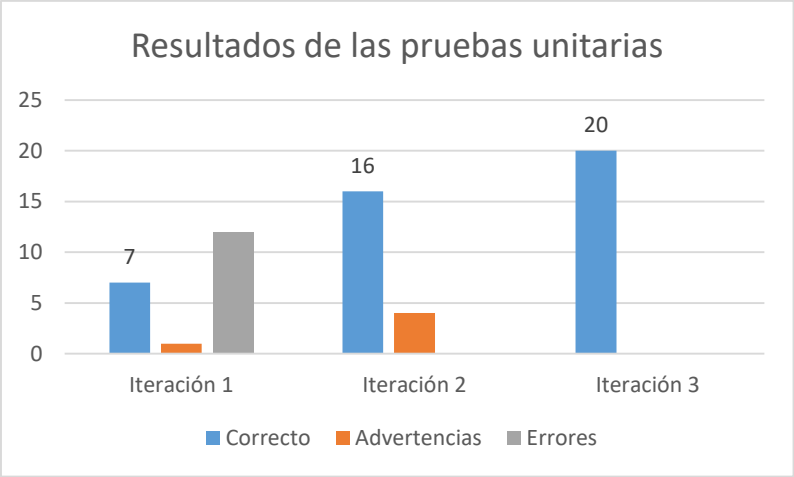

Figura 3.7. Resultados de las pruebas unitarias utilizando *django test suite*.

Adicionalmente y con el objetivo de garantizar una mayor precisión en las pruebas unitarias realizadas se empleó Postman (Figura 3.8), sobre el cual se montaron los *endpoints* de la REST API del sistema, con toda la información necesaria para realizar cada petición HTTP.

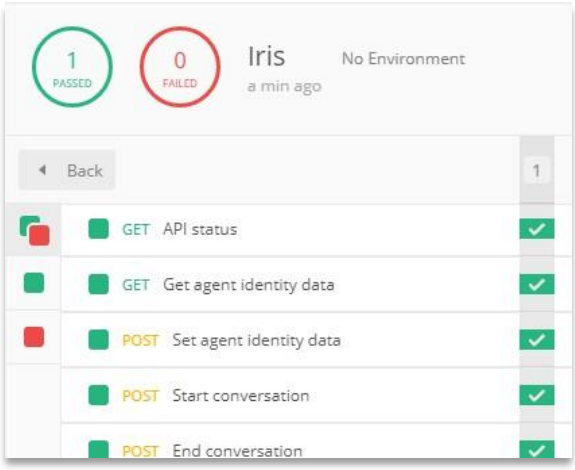

Figura 3.8. Resultados de las pruebas usando Postman.

Como se puede apreciar en la parte superior izquierda de la Figura 3.8, el test se ejecutó correctamente y no falló, por lo cual se puede afirmar que las pruebas unitarias arrojan resultados satisfactorios en este *software*.

3.2.2. Pruebas de aceptación

La disciplina de Pruebas de Aceptación constituye la última prueba que se realiza antes del despliegue del sistema, su objetivo es verificar que el *software* desarrollado esté listo y que puede ser usado por los usuarios finales para

ejecutar aquellas funciones y tareas para los cuales fue desarrollado (Rodríguez Sánchez, 2015).

Diseño de Casos de Prueba

En los Diseños de Casos de Prueba (DCP) se incluyen la descripción de los principales escenarios, actores, posibles entradas, variables que intervienen en el proceso y flujo central donde se realiza el procedimiento. Estos DCP permiten al probador introducir los valores que considere necesarios, ya que están escritos de forma genérica. A continuación, se muestra el CP propuesto para el RF "Incluir red semántica" (Tabla 3.2).

Tabla 3.2. Diseño de caso de prueba "Incluir red semántica"

DCP Incluir red semántica					
Descripción general: El caso de prueba inicia cuando el administrador autenticado se encuentra en la interfaz gráfica de la administración de XAUCE ZERA. **Condiciones de ejecución:** El usuario debe estar autenticado y debe estar en la GUI de la administración de XAUCE ZERA. XAUCE ZERA debe estar configurada para acceder al servidor del STI. El servidor del STI debe estar ejecutándose.					
SC11. Incluir red semántica					
Escenario	**Descripción**	**1**	**2**	**Respuesta del sistema**	**Flujo central**
EC 1.1 Selecciona la opción "incluir"	El administrador autenticado selecciona la opción "incluir" en la página del listado de redes semánticas.	N/A	N/A	Se despliega la ventana del chat y se muestra el primer mensaje del STI saludando al usuario.	Administración/ Redes semánticas
EC 1.2 Guardar los datos	Se introducen los datos: Selecciona el curso al que se	V	V	El sistema envía los datos introducidos al servidor del STI,	Administración/ Redes semánticas/

	le va a asignar una red semántica, así como el archivo de red semántica. Selecciona la opción "subir".			el cual añade la nueva red semántica, asociada al curso seleccionado.	Incluir/ Subir
EC 1.3 Datos incompletos	Se introducen los datos y se selecciona la opción "subir".	I V I	V I I	El sistema señala el o los campos obligatorios que no hayan sido introducidos y muestra el siguiente mensaje para cada uno: "Debe rellenar este campo".	Administración/ Redes semánticas/ Incluir/ Subir

Descripción de las variables

No	Nombre de campo	Clasificación	Valor nulo	Descripción
1	Curso	Cuadro de selección	No	Nombre del curso al cual se le va a asociar la red semántica
2	Red semántica	Examinar	No	Archivo de definición de la red semántica, debe cumplir el formato adecuado y tener extensión “.json”.

3.3. Conclusiones parciales

Partiendo de los artefactos generados en la disciplina “Análisis y Diseño”, se procedió a la implementación de las funcionalidades requeridas por el cliente a través de la utilización de las herramientas y tecnologías seleccionadas.

La implementación del sistema permitió concretar el funcionamiento de cada uno de los módulos del sistema propuesto, siendo la inclusión de comandos, el

compilador y la inclusión de la red semántica los elementos de mayor complejidad.

Se realizaron pruebas unitarias y de aceptación, las cuales arrojaron varias no conformidades que fueron resueltas de forma satisfactoria, logrando con esto verificar el funcionamiento de la solución propuesta.

Conclusiones

El desarrollo de la presente investigación permitió llegar a las siguientes conclusiones generales:

- El análisis realizado del estado del arte fundamentó la necesidad de llevar a cabo el desarrollo de un sistema tutor inteligente de tercera generación para la Plataforma educativa XAUCE ZERA.
- Siguiendo la metodología AUP en su variante UCI, y con las herramientas y tecnologías seleccionadas se logró el diseño, ejecución y pruebas de los módulos clásicos de un STI, con características novedosas en cuanto a representación del conocimiento y procesamiento del lenguaje natural.
- El STI Iris introduce en la Plataforma Educativa XAUCE ZERA el proceso de tutoría en lenguaje natural, tomando en cuenta diferentes dominios de conocimiento, lo cual da cumplimiento al objetivo trazado en la investigación.

Recomendaciones

- Incluir soporte para varios idiomas en base a diferentes modelos lingüísticos, patrones de entrada, respuestas estáticas, entre otros aspectos.
- Brindar retroalimentación cuando el usuario escriba frases de entrada con errores ortográficos.
- Desarrollar una herramienta de autor que permita generar, a partir de una GUI el archivo de red semántica que soporta el producto descrito.

Referencias bibliográficas

(2017). Obtenido de The Andes Physics Tutor: http://www.andestutor.org/

(2018). (Python Software Foundation) Obtenido de Sitio oficial de Python: https://www.python.org/

(2018). (Explosion AI) Obtenido de Sitio oficial de spaCy: https://spacy.io/

(2018). Obtenido de Django Rest Framework: http://www.django-rest-framework.org/

(2018). Obtenido de Django Documentation: https://docs.djangoproject.com/en/2.0/faq/general/#django-appears-to-be-a-mvc-framework-but-you-call-the-controller-the-view-and-the-view-the-template-how-come-you-don-t-use-the-standard-names

(febrero de 2018). Obtenido de Sitio oficial de Git: https://git-scm.com/

Al-Bastami , B., & Abu Naser, S. (enero de 2017). Design and Development of an Intelligent Tutoring System for C# Language. *IV(10)*. EUROPEAN ACADEMIC RESEARCH. Obtenido de https://philpapers.org/rec/ALBDAD-4

Aldahdooh, R., & Abu Naser , S. (enero de 2017). Development and Evaluation of the Oracle Intelligent Tutoring System (OITS). *IV(10)*. EUROPEAN ACADEMIC RESEARCH. Obtenido de https://philpapers.org/rec/ALDDAE-2

Alhabbash, M., Mahdi, A., & Abu Naser, S. (diciembre de 2016). An Intelligent Tutoring System for Teaching Grammar English Tenses. *IV(9)*. EUROPEAN ACADEMIC RESEARCH. Obtenido de https://philpapers.org/rec/ALHAIT

Almurshidi, S., & Abu Naser , S. (enero de 2017). Design and Development of Diabetes Intelligent Tutoring System. *IV(10)*. Obtenido de https://philpapers.org/rec/ALMDAD-2

Almurshidi, S., & Abu Naser, S. (enero de 2017). Stomach disease intelligent tutoring system. *2(1)*, 26-30. International Journal of Advanced Research and Development. Obtenido de https://philpapers.org/rec/ALMSDI

Al-Nakhal, M., & Abu Naser , S. (enero de 2017). Adaptive Intelligent Tutoring System for Learning Computer Theory. *IV(10)*. EUROPEAN ACADEMIC RESEARCH. Obtenido de https://philpapers.org/rec/ALNAIT

Anand, S., & Michael, P. (1995). BDI Agents: From Theory to Practice. *Proceedings of the First International Conference on Multiagent Systems.* Melbourne: AAAI. Obtenido de https://www.aaai.org/Papers/ICMAS/1995/ICMAS95-042.pdf

Arnau, D., Arevalillo-Herráez, M., Puig, L., & González-Calero, J. (2013). Fundamentals of the design and the operation of an intelligent tutoring

system for the learning of the arithmetical and algebraic way of solving word problems.

Azoulay-Schwartz, R., & Hani, Z. (2006). An Intelligent Tutoring System for Efficient Usage of Databases.

Barbhuiya, R., Mustafa, K., & Jabin, S. (junio de 2011). Design specifications for a generic Intelligent Tutoring System. Conference paper. New Delhi, India: Jaima Millia Islamia University. Obtenido de https://www.researchgate.net/publication/232697617

Barr, A., & Feigenbaum, E. (1981). The Handbook of Artificial Intelligence. Volumen I. William Kaufman.

Bellman, R. (1978). An Introduction to Artificial Intelligence: Can Computers Think? Boyd & Fraser Publishing Company.

Bello Pérez, R., García Valdivia, Z., García Lorenzo, M., & Reynoso Lobato, A. (2002). Aplicaciones de la Inteligencia Artificial. *Primera Edición*. México: Universidad de Guadalajara, Centro Universitario de Ciencias Económico Administrativas.

Bruno, R. (2005). Análisis de la percepción de los alumnos y de los docentes para la incorporación de un sistema tutor inteligente como facilitador del aprendizaje.

Buchanan, B., & Shortliffe, E. (1984). Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project. Addison-Weslwy, Reading, MA.

Buchanan, B., Sutherland, G., & Feigenbaum, E. (1969). Heuristic DENDRAL: a program for generating explanatory hypotheses in organic chemistry. *Machine Intelligence 4*, 209-254. Edinburgh, Scotland: Edinburgh University Press.

Burns, H., & Capps, C. (1988). Foundations of STI: An Introduction. *Foundations of Intelligent Tutoring Systems, Lawrence Erlbaum Associates*. (J. J. (editores), Ed.) Polson, M. C., Estados Unidos de América.

Carbonel, J., & Collina, A. (1973). Natural semantics in artificial intelligence. Cambridge, Massachusetts. Obtenido de http://www.ijcai.org/Proceedings/73/Papers/036.pdf

Carbonell, J. (1970). AI in CAI: An artificial intelligence approach to computer assisted instruction. *IEEE transaction on Man Machine System, V11 n.4*, 190-202.

Cataldi, Z., & Lage, F. (marzo de 2009). Sistemas Tutores Inteligentes orientados a la enseñanza para la comprensión. Ciudad de Buenos Aires, Argentina: EDUTEC Revista Electrónica de Tecnología Educativa. Número 28.

Choi, J. D., Tetreault, J., & Stent, A. (2015). It Depends: Dependency Parser Comparison Using A Web-based Evaluation Tool. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing* (págs. 387-396). Beijing: Association for Computational Linguistics. Obtenido de https://aclweb.org/anthology/P/P15/P15-1038.pdf

Clancey, W. (1979). *Dialogue management for rulebased tutorials, Proc: International Joint Conference on AI.* San Francisco, Estados Unidos de América: Morgan Kaufmann Publishers Inc.

Crawford, D. (julio de 2013). *7 Things you should know about intelligent tutoring systems.* Obtenido de EDUCASE Learning Initiative: https://library.educause.edu/resources/2013/7/7-things-you-should-know-about-intelligent-tutoring-systems

Davis, R., Shrobe, H., & Szolovits, P. (1993). What Is a Knowledge Representation? *AI Magazine, 14, 1*. Association for the Advancement of Artificial Intelligence (AAAI). doi:https://doi.org/10.1609/aimag.v14i1.1029

Dede, C. (1986). A review and synthesis of recent research in intelligent computer-assisted instruction. International man-machine studies. 329-353.

Documentación de Django-cors-headers. (2018). Obtenido de https://pypi.org/project/django-cors-headers/

Duarte Correa, Y., & Segovia Vega, R. (junio de 2010). Propuesta de arquitectura para un Sistema Tutor Inteligente de apoyo al proceso de aprendizaje. La Habana, Cuba: Universidad de las Ciencias Informáticas.

Duda, R., Gasching, J., & Hart, P. (1980). Model Design in the Prospector Consultant System for Mineral Exploration. *Expert Systems in the Microelectronic Age*, 153-167. Edinburgh: Edinburgh University Press.

Durango Herández, J., & Pascuas Rengifo, Y. (5 de agosto de 2015). Los sistemas tutores inteligentes y su aplicabilidad en la educación. *Revista de la Facultad de Educación, Ciencias Humanas y Sociales*. Caquetá, Colombia: Corporación Universitaria Iberoamericana. Obtenido de https://dialnet.unirioja.es/descarga/articulo/5455071.pdf

Eguiluz, J. (2011). Desarrollo web ágil con Symfony 2.

El Haddad, I., & Abu Naser, S. (enero de 2017). ADO-Tutor: Intelligent Tutoring System for leaning ADO.NET. *IV(10)*. Obtenido de https://philpapers.org/rec/ELHAIT-2

Eriksson, H., & Penker, M. (2000). *Business Modeling with UML:Business Patterns at Work.* Canadá: John Wiley & Sons, Inc.

Fang, N., & Guo, Y. (2013). A Web-Based Interactive Intelligent Tutoring System for Undergraduate Engineering Dynamics.

Fernández, R., Server, P., & Carballo, E. (Enero de 2006). Aprendizaje con nuevas tecnologías paradigma emergente. ¿Nuevas modalidades de aprendizaje? *EDUTEC, Revista Electrónica de Tecnología Educativa, 20*. Obtenido de http://edutec.rediris.es/Revelec2/revelec20

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (2009). *Design Patterns.* Pearson Education Corporate Sales Division.

Giraffa, L., Nunes, M., & Viccari, R. (1997). Multi-Ecological: an Learning Environment using Multi-Agent architecture. Proc. MASTA'97. Coimbra: DE-Universidade de Coimbra.

González, C. (2004). Sistemas Tutoriales Inteligentres en la educación: una revisión de las líneas de investigación y aplicaciones actuales.

Guardia Robles, B. (1993). Asesores Inteligentes para apoyar al proceso de enseñanza de lenguajes de programación. Tesis de grado. Instituto Tecnológico de Monterrey (ITESM), México.

Hebb, D. (1949). The Organization of Behavior. New York: Jhon Wiley & Sons.

Hilles, M., & Abu Naser , S. (enero de 2017). Knowledge-based Intelligent Tutoring System for Teaching Mongo Database. *IV(10)*. EUROPEAN ACADEMIC RESEARCH. Obtenido de https://philpapers.org/rec/HILKIT

IEEE. (18 de julio de 2017). *The 2017 Top Programming Languages*. Obtenido de IEEE Spectrum: https://spectrum.ieee.org/computing/software/the-2017-top-programming-languages

Introducing JSON. (2018). Obtenido de https://www.json.org/

Jacobson, G., & James, R. (2000). *El Lenguaje Unificado de Modelado. Manual de referencia.* Madrid: Pearson Education.

Joskowicz, J. (2008). *Reglas y prácticas en eXtreme Programming* (Vol. 22). Universidad de Vigo.

Larman, C. (2003). UML y Patrones. Una introducción al análisis y diseño orientado a objetos y al proceso unificado. Segunda edición. Madrid: PEARSON EDUCATION, S.A.

Liddy, E. (2001). Natural Language Processing. *Encyclopedia of Library and Information Science, 2nd Ed*. New York, United States of America: Marcel Decker, Inc. Obtenido de http://surface.syr.edu/cgi/viewcontent.cgi?article=1019&context=cnlp

Litman, D. (2016). Natural Language Processing for Enhancing Teaching and Learning. *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*.

Manual de PHP. (2018). Obtenido de http://www.php.net/manual/en/introwhatcando.php

Matplotlib official site. (30 de 05 de 2018). Obtenido de https://matplotlib.org/

Minsky, M. (1954). Neural Nets and the Brain-Model Problem (SNARC). Tesis de Doctorado. New Jersey, Estados Unidos: Universidad de Princeton.

Mitrovic, A. (1998). A Knowledge-Based Teaching System for SQL. Obtenido de https://pdfs.semanticscholar.org/2608/87ecabceaec1eff6c04d964c0ffed264aca6.pdf

Networkx. (05 de 30 de 2018). Obtenido de Github: https://networkx.github.io/

Ossandón, Y. (2001). Sistemas Tutores: Una alternativa para el proceso enseñanza-aprendizaje en la ingeniería.

Parra, F. (2010). Sistema Tutorial Inteligente.

Pérez González, D., & Nuñez Padrón, S. (2015). SCORM View: Visor de paquetes SCORM para XauceMóvil. Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas. La Habana: Universidad de Ciencias Informáticas.

Picard, R. (1997). Affective Computing. *M.I.T Media Laboratory Perceptual Computing Section Technical Report No. 321*. MIT Press. Obtenido de affect.media.mit.edu/pdfs/95.picard.pdf

PostgreSQL, Portal en español sobre PostgreSQL. (2018). Obtenido de http://www.postgresql.org.es/sobre_postgresql

Pressman, R. (2010). *Ingeniería del Software: Un Enfoque Práctico* (7ma ed.). McGraw-Hill.

Rational Rose. (s.f.). Obtenido de http://es.scribd.com/doc/74347179/Rational-Rose

Rodríguez Sánchez, T. (2015). Metodología de desarrollo para la Actividad produciva de la UCI. La Habana, Cuba: Universidad de las Ciencias Informáticas.

Rodríguez, A., Castillo, J., & Lira, A. (2013). Diseño de un Sistema Tutorial Inteligente.

Ruiz, F. (2000). *Modelo Entidad/Relación.*

Shortliffe, E. (1976). *Computer based medical consultations, MYCIN.* New York, Estados Unidos de América.

Sitio oficial de Net Beans. (2018). Obtenido de http://netbeans.org

Sitio oficial de Postman. (2018). Obtenido de https://www.getpostman.com/

Sitio oficial de Visual Paradigm. (2018). Obtenido de http://visual-paradigm.com/

Slagle, J. (1963). A heuristic program that solves symbolic integration problems in freshman calculus. *Journal of the Association for Computing Machinery*. Feldman.

Sleeman, D. (1987). PIXIE: a shell for developing intelligent tutoring systems. *AI & education: Learning environments and intelligent tutoring systems*, 239-265.

Sommerville, I. (2005). *Ingeniería de Software Séptima Edición.* Madrid: Pearson Education.

Staab, S., & Studer, R. (Edits.). (2009). *Handbook of Ontologies* (Segunda Edición ed.). Springer-Verlag Berlin Heidelberg. doi:10.1007/978-3-540-92673-3

Standard ECMA-404. The JSON Data Interchange Syntax. (diciembre de 2017). *2da Edición*. ECMA International. Obtenido de http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf

Stuart, J., Norvig, P., Canny, J., Malik, J., & Douglas, D. (1995). Artificial Intelligence A Modern Approac. Prentice Hall, Englewood Cliffs, New Jersey 07632., Estados Unidos de América.

ThoughtWorks. (2016). Technology Radar. *16*. ThoughtWorks. Recuperado el 28 de febrero de 2018, de https://www.thoughtworks.com/radar

ThoughtWorks. (2017). Technology Radar. *17*. ThoughtWorks. Recuperado el 27 de febrero de 2018, de https://www.thoughtworks.com/radar

VanLehn, K. (1988). Student Modeling. *M. Polson. Foundations of Intelligent Tutoring systems*, 55-78. Hillsdale. N.J. Lawrence Erlbaum Associates.

VanLehn, K., Lynch, C., Schulze, K., Shapiro, J., Shelby, R., Taylor, L., . . . Wintersgill, M. (2005). The Andes Physics Tutoring System: Lessons Learned, 15(3). *International Journal of Artificial Intelligence in Education*. Obtenido de https://oli.cmu.edu/wp-content/uploads/2012/05/VanLehn_2005_Andes_Physics_Tutoring_System.pdf

W3C. (2018). Obtenido de Sitio oficial de la World Wide Web Consortium (W3C): https://www.w3.org

Wenger, E. (1987). *Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge.* Morgan Kaufmann, Estados Unidos.

Wijekumar, K., Meyer, B., & Lei, P. (2013). High-fidelity implementation of web-based intelligent tutoring system improves fourth and fifth graders content area reading comprehension.

Wolf, B. (1984). Context Dependent Planning in a Machine Tutor. Ph.D. Dissertation. University of Massachusetts, Amherst, Massachusetts.

Yang, A., Leung, H., Yue, L., & Deng, L. (febrero de 2013). Generating a two-phase lesson for guiding beginners to learn basic dance movements. *Computers and Education, 61*. doi:10.1016/j.compedu.2012.09.006